ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Técnicas de Diseño y Optimización aplicadas a Filtros de Radiofrecuencia para los Futuros Servicios de Telefonía Móvil

AUTOR: Manuel Francisco Jiménez Nogales
DIRECTOR: Alejandro Alvarez Melcón
CODIRECTORES: Fernando Quesada Pereira
David Cañete Rebenasque

Septiembre/2007

Índice

1. Introducción

1.1. Introducción y objetivos

Los sistemas de microondas tienen múltiples usos en la vida moderna: comunicaciones vía satélite, radio telefonía móvil, radar, etc. Los filtros de microondas y radiofrecuencia se usan en todos estos sistemas para separar la señal deseada del resto. Un filtro es un bloque que deja pasar solo la potencia que se encuentra en un rango determinado del espectro; las señales que se encuentran fuera de dicho rango son reflejadas. El diseño de filtros parte de la síntesis de circuitos a partir de especificaciones técnicas. Esta síntesis proporciona una red prototipo desde la que desarrollar distintos tipos de realizaciones de microondas, como líneas de transmisión TEM (Transversal Electro-Magnética), guíaondas, microstrip o dieléctricos resonantes.

Un sistema es resonante cuando presenta un comportamiento selectivo para algunas frecuencias. En general todas las estructuras periódicas presentan resonancia, por ejemplo la existencia de resonancias en los cristales dieléctricos se debe a la estructura periódica en que se disponen los átomos del cristal. Esta propiedad que presentan las estructuras periódicas es explotada para realizar filtros en el espectro de las microondas. Una de las metodologías más utilizadas consiste en instalar obstáculos idénticos en el interior de una guía de onda. Como resultado, la señal que viaja por la guía de onda será filtrada en una frecuencia relacionada con la distancia entre obstáculos.

El sistema GSM utiliza TDMA (Acceso Múltiple por División en el Tiempo), por lo que la estación base y el receptor móvil transmiten simultáneamente. El propósito principal del filtro es proteger el Amplificador de Bajo Ruido (LNA) y el mezclador posterior de señales no deseadas que debido a su potencia o frecuencia perjudiquen el correcto funcionamiento.

Las especificaciones para estos filtros suelen ser bastante severas, cerca de los límites teóricos en términos de selectividad de frecuencias y distorsión de fase. Los filtros prototipo sólo sirven como aproximación, ya que rara vez cumplen dichas especificaciones, haciendo necesaria una optimización de su diseño. Existen muchas técnicas de optimización aplicables a cualquier problema, generalmente relacionadas con encontrar el mínimo o el máximo de una función matemática. Para el caso de los filtros se ha creado una función que los evalúa: su ancho de banda, selectividad, atenuación, etc. Las variables de la función son las dimensiones de los distintos elementos que lo componen. Si encontramos el punto para el cual la función sea máxima, tendremos las dimensiones del filtro que mejor cumple con las especificaciones. Se ha elegido como procedimiento para la optimización los Algoritmos Genéticos, por su versatilidad para trabajar con muchas variables, y sobre funciones con orografía compleja.

Se ha creado un programa en **MatLab** que facilita el proceso de diseño de filtros. Este programa consta de una Interfaz Gráfica de Usuario (GUI) desde la que se introducen las especificaciones del filtro que queremos generar. El programa creará un primer prototipo siguiendo el modelo de Chebyshev. A partir de las diferencias entre la respuesta del filtro prototipo y del que buscamos, elegimos las variables (dimensiones) a optimizar y configuramos el algoritmo genético para que realice la búsqueda del filtro optimizado. En este proceso de diseño, la herramienta hace uso internamente del programa **FEST 2.0** para calcular la respuesta en frecuencia de los filtros. Finalmente, se presenta un resultado. El programa **Fest** es

una herramienta creada por la Agencia Espacial Europea para ayudar a la industria al desarrollo de filtros para aplicaciones por satélite. La figura siguiente muestra las funciones que desempeña cada programa dentro del proceso común de optimización, y la interrelación entre ambos:

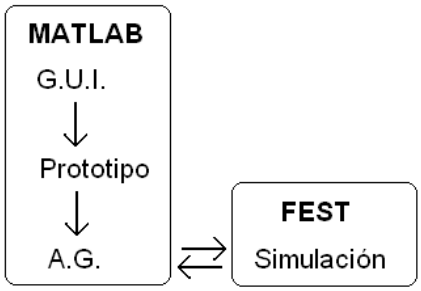

Figura 1. Transferencia entre Matlab y Fest

En este informe primero se presentan los aspectos teóricos en los que se basa la herramienta desarrollada y luego se explica el funcionamiento de la misma acompañando con ejemplos reales de diseño de filtros.

1.2. Filtro Paso Bajo Ideal

Un filtro paso bajo ideal [3] es una abstracción teórica no realizable en la práctica pero que resulta muy útil en el proceso de diseño de filtros. Consiste en un filtro paso bajo sin pérdidas en la banda de paso y sin banda de transición. Así

$$|H(j\omega)| = 1 \; |\omega| < \omega_c \tag{1}$$

$$|H(j\omega)| = 0 \; |\omega| > \omega_c \tag{2}$$

La respuesta de fase es lineal en la banda de paso, por lo que el retardo de grupo será constante y no habrá distorsión de fase, lo cual podría aumentar la tasa de errores de bit (BER) en el sistema.

$$\psi(\omega) = -k\omega \tag{3}$$

$$T_g = \frac{-d\psi(\omega)}{d\omega} = k \tag{4}$$

Si combinamos ambas condiciones (eliminación total de señales de la banda de rechazo y ninguna distorsión de amplitud o de fase en la banda de paso), tenemos que la función de transferencia del filtro ideal es:

$$H(j\omega) = \exp(-jk\omega) \quad |\omega| < \omega_c \tag{5}$$

$$H(j\omega) = 0 \quad |\omega_c| > \omega_c \tag{6}$$

Analizamos la respuesta al impulso:

$$h(t) = \frac{1}{2\pi} \int H(j\omega) exp(j\omega t) d\omega \tag{7}$$

$$h(t) = \frac{1}{\pi} sinc(t-k) \tag{8}$$

La figura siguiente representa la función *sinc* retardada k segundos:

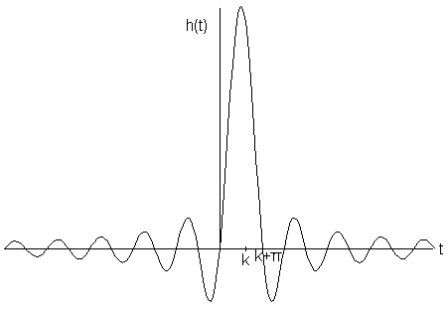

Figura 2. $Sinc(t-k)$

Este retraso de la señal corresponde al *retardo de grupo*.

El prototipo paso-bajo es una red de dos puertos con frecuencia angular de corte $\omega = 1$, carga de 1Ω y alimentada por un generador cuya resistencia también es de 1Ω.

Una red sin pérdidas contiene sólo elementos reactivos. Estos elementos reactivos resuenan a las frecuencias de la banda de paso, y reflejan el resto de frecuencias

(banda de rechazo). Si bien todo filtro de microondas tendrá características resistivas, el diseño de una teórica red sin pérdidas simplifica el proceso de desarrollo del filtro final. Siendo $Z(\omega)$ la impedancia de entrada de la red:

$$Z(s) = \frac{V(s)}{I(s)} = R(\omega) + jX(\omega) \tag{9}$$

El coeficiente de reflexión es

$$\Gamma(s) = \pm\frac{Z(s) - 1}{Z(s) + 1} \tag{10}$$

Siendo $R(s) \geq 0 \rightarrow |\Gamma| \leq 1$.

Suponiendo que se trate de un sistema lineal, invariante en el tiempo, pasivo y sin pérdidas, tendremos que $R(s) = 0$. Si expresamos la impedancia de entrada como una fracción:

$$Z(s) = \frac{N(s)}{D(s)} = \frac{a_n + b_n}{a_d + b_d} \tag{11}$$

Representando a a la parte par y b a la impar de cada polinomio. La parte par de la impedancia es real y la impar imaginaria. $R(s) = 0$ implica que $Z(s)$ sea impar e imaginario:

$$\frac{Z(s) + Z(-s)}{2} = 0 \tag{12}$$

$$\frac{a_n + b_n}{a_d + b_d} + \frac{a_n - b_n}{a_d - b_d} = 0 \tag{13}$$

$$\frac{a_n}{b_n} = \frac{b_d}{a_d} \tag{14}$$

Por tanto:

$$Z(s) = \frac{b_n}{a_d} = \frac{a_n}{b_d} \tag{15}$$

Deducimos que $Z(s)$ es la razón entre un polinomio impar y otro par. Esto es lo que se conoce como **función reactiva**. Estas funciones tienen una solución general de la forma:

$$Z(s) = A_\infty s + \frac{A_0}{s} + \sum_{i=1}^{m} \frac{2A_i s}{s^2 + \omega_i^2} \tag{16}$$

Siendo en este caso $Z(j\omega) = jX(\omega)$ concluimos:

$$X(\omega) = A_\infty \omega - \frac{A_0}{\omega} + \sum_{i=1}^{m} \frac{2A_i \omega}{\omega_i^2 - \omega^2} \tag{17}$$

1.3. Diseño de redes escalonadas

Una red escalonada [3] se corresponde con el esquema de la figura siguiente, donde las impedancias Z son elementos reactivos:

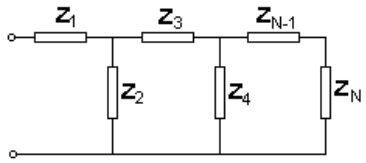

Figura 3. Esquema de impedancias de una red escalonada

Existen varios métodos para sintetizar una red escalonada que responda a una función de impedancia de entrada dada, como el **método de síntesis Darlington**. Darlington demostró que cualquier función positiva y real puede ser vista como la impedancia de entrada de una red de dos puertos pasiva y sin pérdidas, terminada en una carga. Sea $Z(s)$ la impedancia de entrada a sintetizar. Vamos a ir evaluando los sucesivos residuos de Z cuando $s \to \infty$ para determinar los valores de las impedancias Z_i de la red:

$$Z_1 = \frac{Z(s)}{s}|_s \to \infty \tag{18}$$

$$Z_a(s) = Z(s) - Z_1 s \tag{19}$$

Siendo Z_a la impedancia que se tiene del resto de la red a la derecha de Z_1.

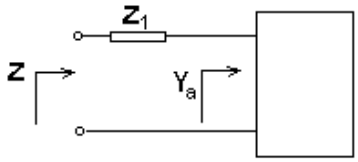

Figura 4. Relación entre Z , Z_1 y Z_a

Para calcular la segunda impedancia tomaremos su admitancia, ya que está en paralelo:

$$Y_2 = \frac{Y_a(s)}{s}|_s \to \infty \tag{20}$$

$$Y_b(s) = Y_a(s) - Y_2 s \tag{21}$$

Con este método sucesivo calculamos los valores de todos los elementos de la red.

1.4. La matriz S

Para caracterizar una red de dos puertos podemos utilizar distintos parámetros que reflejen sus características y propiedades, además de servir para simplificar su análisis. El método que seguiremos aquí es el de la **matriz de Scattering**.

Sea una red de dos puertos (cuadripolo) como la de la figura:

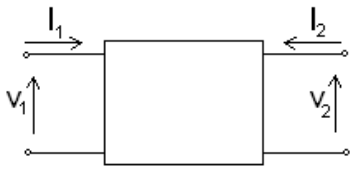

Figura 5. Cuadripolo

Esta red se describe por la relación entre la corriente y el voltaje a la entrada y a la salida.

$$[V] = [Z][I] \tag{22}$$

$$V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \quad I = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \tag{23}$$

Descomponiendo V e I de la manera siguiente:

$$[V] = [a] + [b] \qquad [I] = [a] - [b] \tag{24}$$

Donde a y b representan respectivamente las señales incidentes y reflejadas en ambos puertos. Tenemos que la matriz S relaciona ambos vectores auxiliares:

$$[b] = [S][a] \quad \rightarrow \quad \begin{bmatrix} S_{11} S_{12} \\ S_{21} S_{22} \end{bmatrix} \tag{25}$$

De la ecuación 22 deducimos que:

$$[Z] = \frac{1 + [S]}{1 - [S]} \tag{26}$$

- S_{11} es el coeficiente de reflexión a la entrada.
- S_{12} es el coeficiente de reflexión de la salida.
- S_{21} es el coeficiente de transmisión directa.
- S_{22} es el coeficiente de transmisión inversa.

Si expresamos S_{12} en decibelios tenemos las **pérdidas de inserción** L_A, que miden la atenuación que sufre la señal dentro del filtro. Igualmente, si medimos

S_{11} en decibelios obtenemos las **pérdidas de retorno** L_R, que miden la bondad del acoplamiento de la red con la carga posterior:

$$L_A(dB) = -20\log|S_{12}(j\omega)| \tag{27}$$
$$L_R(dB) = -20\log|S_{11}(j\omega)| \tag{28}$$

En las redes sin pérdidas se cumple la condición de unitariedad:

$$|S_{11}|^2 + |S_{12}|^2 = 1 \tag{29}$$

Esta propiedad dice que la energía que no se transmite a la salida se refleja por la entrada, lo que permite deducir que en una red sin pérdidas perfectamente acoplada, las pérdidas de inserción serán nulas y las de retorno infinitas.

El retardo de grupo que produce el filtro se refleja en la fase de S_{21}. Interesa que la respuesta en fase de este parámetro sea lo más lineal posible en la banda de paso; pero, por contra, cuanto más selectivo sea el filtro mayor será esta distorsión de fase, por lo que hay que alcanzar un compromiso de diseño que satisfaga ambas características.

$$T_g(\omega) = \frac{-d\angle S_{21}(\omega)}{d\omega} \tag{30}$$

1.5. Diseño del prototipo paso bajo

Al idear un filtro partimos de unos requisitos que debe cumplir en cuanto a selectividad de frecuencia, atenuación máxima permitida en la banda de paso y mínima para la banda de rechazo. Existen diversos métodos a la hora de diseñar un filtro; aquí seguiremos el método de las pérdidas de inserción. El primer paso para diseñar este filtro consiste en transformar estas especificaciones para crear un prototipo paso bajo. Un prototipo paso bajo es una red de dos puertos pasiva, recíproca y sin pérdidas, en el que la frecuencia de corte está normalizada $\omega_c = 1$, así como la impedancia de carga $\Omega = 1$.

Una vez que tengamos las especificaciones transformadas para el prototipo, podemos diseñarlo. Existen diversas realizaciones de estos prototipos, dependiendo de la forma de la función de transferencia. Pueden seguir la estructura de Butterworth, Chebyshev o elípticos. Este prototipo puede construirse con elementos reactivos (bobinas y condensadores). Para obtener los valores de estos elementos se recurre a tablas normalizadas que tienen los coeficientes de los polinomios que implementan la función deseada. Por último hay que realizar una desnormalización de estos valores para determinar los valores reales de los elementos físicos del filtro, ya sean longitudes de una guíaonda, inversores, stubs, etc.

Existen diferencias entre los filtros diseñados siguiendo polinomios de la forma de Butterworth, Chebyshev o elípticos. En los primeros no exite rizado en la banda de paso, pero el tránsito entre la banda de paso y la de corte es más suave que en los otros dos; los polinomios elípticos consiguen una selectividad mayor, pero presentan rizado en la banda de paso, lo cual distorsiona la señal filtrada. Los filtros de Chebyshev tiene un rizado menor, pero no son tan selectivos.

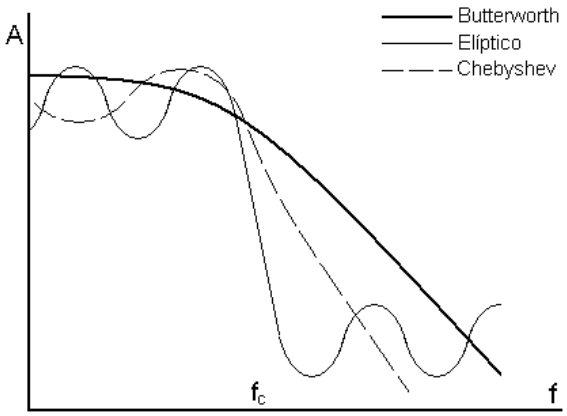

Figura 6. Comparación filtros Butterworth, Chebyshev y elípticos

Las frecuencias de microondas se pueden propagar por líneas de transmisión. Manipulando las dimensiones físicas de estas líneas obtendremos los elementos resonantes que conforman el filtro. La ventaja de este tipo de filtros es su sencilla fabricación, porque no requieren componentes extras o soldados. Sin embargo, los efectos electromagnéticos que surgen a esas frecuencias son muy complejos y, para su diseño, se requiere un proceso de optimización mediante simulador software.

1.6. FEST 2.0

FEST 2.0 es una herramienta desarrollada por la Agencia Espacial Europea para el diseño y análisis de guías de onda rectangulares con discontinuidades inductivas y capacitivas en la banda de frecuencias de microondas. Permite incluir gran variedad de elementos, como uniones en T o en Y, stubs terminados en cortocircuito o codos en U y en S, además de las discontinuidades de guíaonda. Realiza un barrido en frecuencia del comportamiento de cualquier combinación de estos elementos para obtener los parámetros **S** de la estructura resultante, así como la respuesta en fase.

Los filtros que vamos a utilizar son guías de onda con escalones inductivos, como muestra el esquema siguiente:

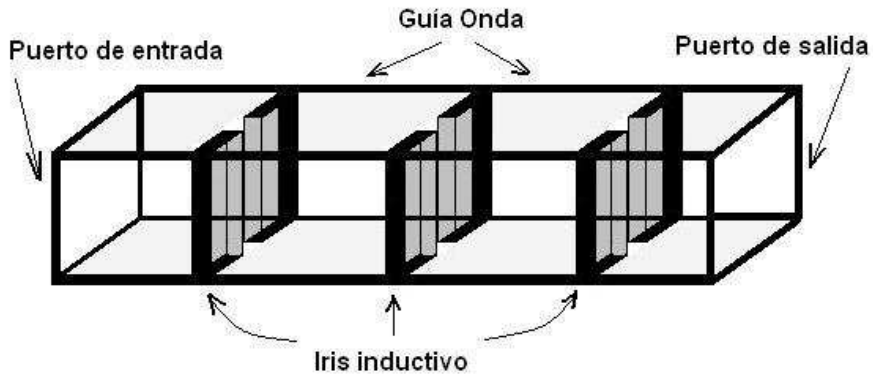

Figura 7. Filtro con írises inductivos

Los elementos de **Fest 2.0** que se han utilizado para formar las estructuras de estos filtros son:

- Puerto de entrada:

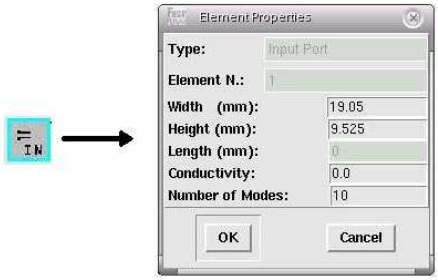

Figura 8. Puerto de entrada

- Puerto de salida:

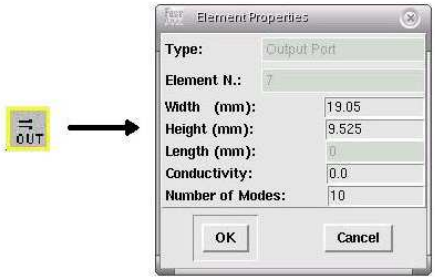

Figura 9. Puerto de salida

- Guías de Onda:

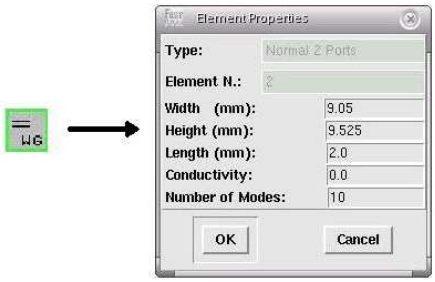

Figura 10. Sección de Guía Onda

- Saltos inductivos:

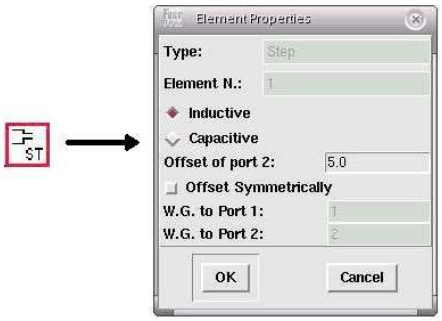

Figura 11. Salto inductivo

Los parámetros de entrada del programa son las dimensiones de cada uno de los elementos que forman la estructura a analizar: anchura, altura y longitud; los saltos se definen mediante su tipo: inductivo o capacitivo.

Las especificaciones de diseño del filtro determinan su orden, o número de escalones inductivos y secciones de guía que lo compondrán. Para aumentar las posibilidades de diseño se permite el uso de filtros de hasta orden 4º. Estas son las plantillas que se han utilizado para los filtros de distinto orden:

- Filtro de orden 1:

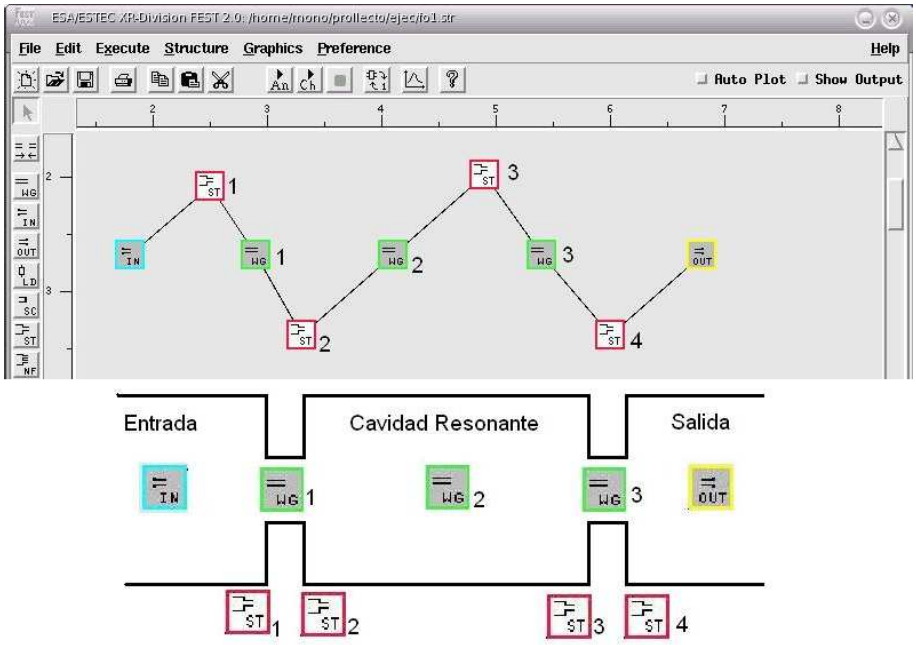

Figura 12. Filtro de orden 1

- Filtro de orden 2:

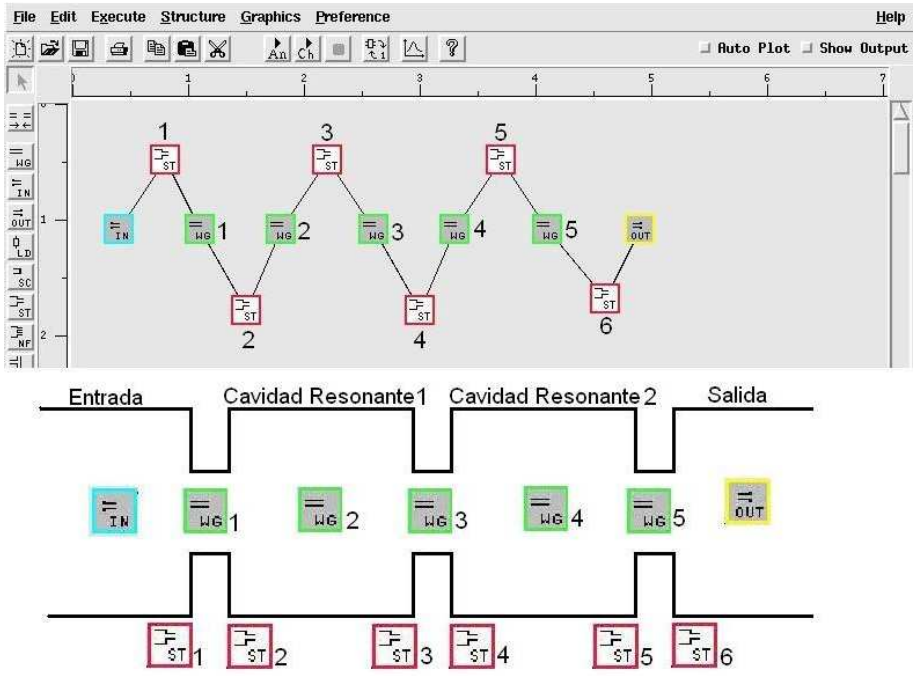

Figura 13. Filtro de orden 2

- Filtro de orden 3:

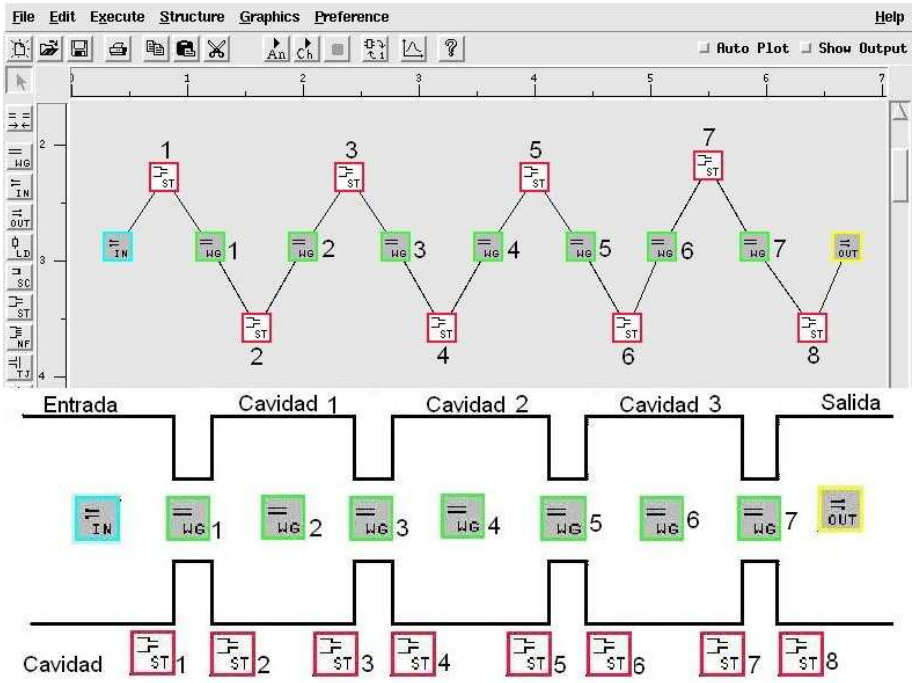

Figura 14. Filtro de orden 3

- Filtro de orden 4:

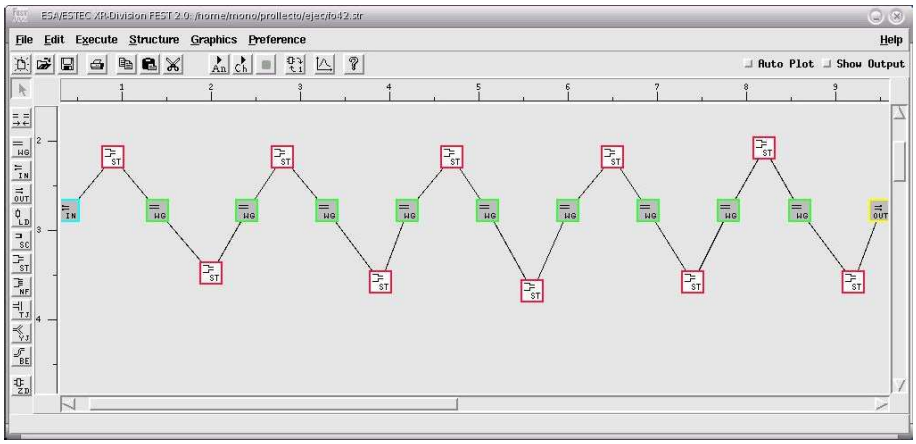

Figura 15. Filtro de orden 4

Partiendo de estas plantillas, se diseñan los filtros que requieran los algoritmos genéticos, modificando las dimensiones de los distintos elementos que los forman, y posteriormente se evalúan.

2. Algoritmos Genéticos

Un algoritmo genético es un procedimiento de optimización que imita los mecanismos de selección y evolución de la naturaleza. El proceso es simple: las características genéticas del individuo determinan su grado de adaptación al medio; los individuos mejor adaptados tendrán más posibilidades de procrear, transmitiendo así su información genética a la siguiente generación. Las combinaciones genéticas que produzcan individuos poco adecuados al medio se desechan automáticamente, porque dichos individuos tienen muy pocas probabilidades de ser seleccionados para formar la siguiente generación. De esta forma, la segunda generación tiene más probabilidades de estar mejor adaptada al medio que la primera.

Los algoritmos genéticos se emplean para resolver distintos problemas en entornos muy diversos, tanto en el campo de la ingeniería como en predicciones financieras o el diseño de fármacos. Son útiles para resolver problemas complejos, en los que intervengan muchas variables y no esté muy claro el camino que hay que seguir para llegar a la solución.

En general, cualquier modalidad de algoritmo genético sigue unos pasos que se repiten iterativamente hasta encontrar una solución aceptable del problema a resolver [2]. Estas fases del algoritmo son:

- *Inicialización*: Lo primero que hay que hacer es crear la población inicial de individuos, la primera generación.
- *Evaluación*: El segundo paso consiste en evaluar los individuos de la población y asignarles una puntuación. Cada individuo representa una combinación determinada de los distintos parámetros e incógnitas del problema. dicha combinación dará un resultado más o menos aceptable del problema que se quiere resolver; dependiendo de la bondad de esta solución se le asigna una puntuación.
- *Comprobación*: Una vez evaluadas las combinaciones propuestas en la población, se comprueba si se ha alcanzado una solución satisfactoria. También se puede dar por finalizado el proceso siguiendo otros criterios, como un número máximo de iteraciones. En caso contrario hay que continuar con el algoritmo.
- *Selección*: Para formar una nueva generación hay que escoger los mejores individuos de la generación actual. Esto se hace de acuerdo con la puntuación que han obtenido en la fase de evaluación. En la naturaleza, los individuos mejor dotados o adaptados al medio tienen más probabilidades de procrear y transmitir sus genes a la siguiente generación. De esta manera se eliminan aquellas configuraciones genéticas más desfavorables. Para elaborar un nuevo individuo harán falta por lo menos dos progenitores.
- *Nueva generación*: A partir de los individuos seleccionados, se obtiene una nueva generación que sustituya a la actual. Esta nueva generación tiene más probabilidades de adaptarse mejor al problema en estudio que su predecesora, ya que se ha constituido a partir del material genético de los individuos más favorables. Llegados a este punto, hay que volver al punto segundo para evaluar la nueva generación y continuar iterando el algoritmo hasta su final.

En la figura siguiente se muestra el diagrama de flujo propio de un algoritmo genético, donde se relacionan las distintas fases del mismo:

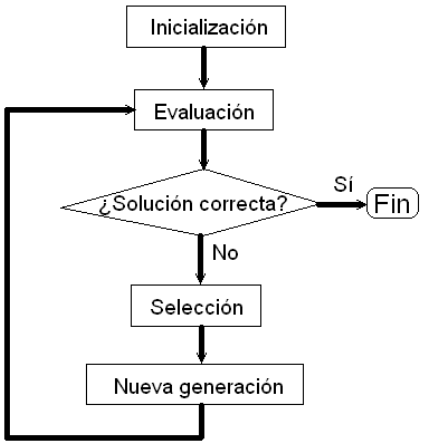

Figura 16. Diagrama de flujo de los Algoritmos Genéticos

La selección natural favorece que los individuos cuya combinación genética sea más apropiada al medio en el que viven lleguen a edad adulta y sean los encargados de procrear. Si los hijos de estos individuos tienen características similares, tendrán más posibilidades de sobrevivir que si sus características fuesen independientes de sus progenitores. Es por esto que se heredan los genes de una generación a otra. En algunas especies la reproducción consiste en crean clones de sus individuos para formar otros nuevos pero idénticos; el problema es que así no evoluciona la especie y, por tanto, este modelo no nos interesa para resolver problemas. Para conseguir individuos ligeramente diferentes a los que ya tenemos y poder así evolucionar una especie, necesitamos combinar por lo menos dos individuos para formar uno nuevo. Es lo que se conoce como reproducción sexual, en la que se combina la información genética de dos individuos para formar otros nuevos y originales.

Es importante introducir un porcentaje pequeño de información genética nueva independiente de los progenitores, que es lo que se conoce como mutación. Las mutaciones son importantes porque proporcionan combinaciones genéticas inéditas, ampliando el espacio de soluciones bajo estudio. A veces son claves para encontrar la solución deseada, pero en otros casos los individuos mutados resultan peores que los originales. No hay que olvidar que en la naturaleza, las mutaciones han dado lugar a los saltos evolutivos más importantes. Las mutaciones también son interesantes en casos de estancamiento de la población bajo estudio; cuando la mayoría de los individuos son muy parecidos entre sí, sus descendientes también lo serán, por lo que se necesita introducir mutaciones para conseguir cambios significativos en los individuos.

2.1. Poblacion inicial

Vamos a analizar ahora más profundamente las distintas fases de que consta un algoritmo genético, para ver las distintas opciones que surgen. La primera fase, como ya vimos, es la de inicialización de la población. A partir del prototipo teórico desarrollado anteriormente, hay que dar un margen de flexibilidad para las distintas variables que se quieran optimizar, creando así el espacio de posibles soluciones. La población inicial podrá cubrir dicho espacio aleatoria o uniformemente. En la figura siguiente se muestra un ejemplo de población inicial uniformemente distribuida para el caso de dos dimensiones o variables a optimizar. El punto central representa el valor teórico de dichas variables, que recordemos que se trata de un valor aproximado

basado en un modelo simplificado del filtro real.

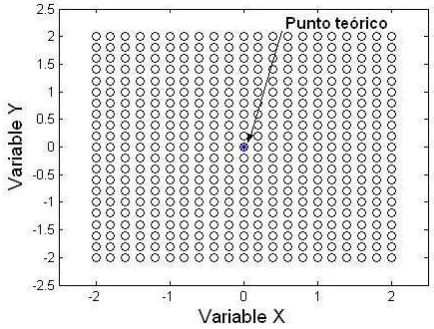

Figura 17. Población inicial uniformemente distribuida para 2 variables

En principio, podría pensarse que una distribución uniforme cubre mejor el espacio de soluciones que si se hace aleatoriamente, pero a veces es mejor introducir cierta aleatoriedad: nótese que en el ejemplo de la figura anterior todos los valores son pares (divididos entre 10), por lo que ninguna combinación genética de ellos dará un valor impar (por ejemplo: (x,y)=(1.1,-0.7)). En este caso, si el número de bits con que codificamos esta información es tal que el intervalo de cuantificación sea menor de 0.2 u.n.[1] estaríamos desaprovechando un bit, el menos significativo, porque tedría el mismo valor para toda la población. Para evitar este tipo de problemas, es mejor crear primero la población inicial codificada en bits, es decir, su información genética, y luego relacionarla con las dimensiones físicas que represente; también es importante dimensionar la población de acuerdo al número de bits empleados.

2.2. Evaluación

En esta fase se relaciona la información genética de cada individuo de la población con las variables físicas que representan, y se analiza el modelo resultante. El objetivo es asignar una puntuación a cada individuo propuesto para así poder compararlos posteriormente y seleccionar los mejores. Esta puntuación tendrá que ser una ponderación de las distintas características que determinan la bondad de un filtro, comparadas con las del filtro ideal. Las puntuaciones se mantendrán dentro de un intervalo de valores, procurando que se cubra lo más ampliamente posible, de manera que la puntuación de los peores elementos de la población esté relativamente distanciada de la de los mejores. Cuando traducimos el código genético de un individuo a las variables asociadas, puede suceder que el resultado sea físicamente incoherente, por lo que debemos desechar dicho individuo.

Observando las puntuaciones obtenidas se puede ver la evolución del algoritmo a lo largo de generaciones, si obtiene mejores resultados que al principio o si no responde bien.

2.3. Comprobación

Una vez analizados todos los individuos de la población hay que comprobar si hemos alcanzado una solución satisfactoria, un filtro que se ajuste a las características que buscamos; en ese caso ya no es necesario continuar iterando el algoritmo,

[1] u.n: unidades normalizadas respecto de una dimensión física determinada

dándolo por concluido. Existen otros criterios para detener el algoritmo en este punto, como puede ser el hecho de haber realizado un número determinado de iteraciones aunque no se haya encontrado la solución requerida, o que la población se haya estancado. Una población se estanca cuando la mayoría de sus miembros son iguales entre sí, porque no se pueden crear individuos distintos a partir de progenitores idénticos genéticamente; esto provoca que la siguiente generación sea igual a la anterior.

2.4. Selección

Para formar una nueva generación se seleccionan los individuos mejor dotados para resolver el problema, desechando así aquellos cuya combinación genética no resulta útil. Esta selección se lleva a cabo a partir de la puntuación que se ha asignado a cada individuo en la fase de evaluación. Existen diversos criterios para realizar la selección: deterministas y estadísticos. El método determinista consiste en escoger directamente los individuos con mejor puntuación; en los métodos probabilísticos se realizan competiciones entre dos aspirantes al azar para elegir sólo uno de ellos. Estas competiciones pueden ser comparaciones directas (el que tenga mayor puntuación gana) o donde intervenga el azar. Analicemos algunos de estos métodos:

- *Ruleta*: Consiste en asignar a cada individuo de la población una probabilidad proporcional a su calificación y elegir por sorteo el ganador. Este procedimiento debe su nombre al paralelismo que guarda con el juego de la ruleta; en este caso, a cada elemento de la población le corresponde un sector de la ruleta cuyo arco esté en proporción con su puntuación. Ver figura 18:

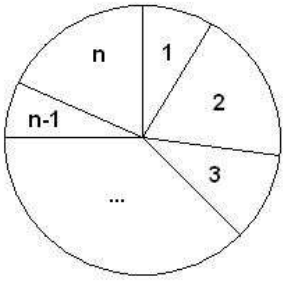

Figura 18. Ruleta de n individuos

- *Torneo*: En un torneo se eligen al azar dos individuos cualesquiera y se selecciona el que tenga mayor puntuación. Este método permite que algunos individuos con poca puntuación sean seleccionados en lugar de otros a priori mejores, pero se gana en diversidad genética.

- *Torneo probabilístico*: Ahora el torneo no es determinista; de los dos preseleccionados para el torneo no gana necesariamente el de mayor puntuación, sino que se realiza un sorteo entre los dos ponderando sus probabilidades de ganar según su puntuación. Esta ponderación puede ser lineal o seguir otros modelos más apropiados cuando las diferencias de puntuación pueden ser grandes. El torneo probabilístico permite que ante torneos iguales o similares se introduzcan individuos que de otra manera serían rechazados. Sean dos individuos, a y b, tales que:

$$f(a) > f(b) \tag{31}$$

Siendo $f(x)$ la función de evaluación. Con esta situación, en un torneo saldría elegido a. Sin embargo, en un torneo probabilístico se introduce un número aleatorio r:

$$0 < r < 1 \tag{32}$$

Este número se compara con una función p que depende de la distancia entre las puntuaciones, determinando qué individuo es seleccionado:

$$\begin{aligned} r > p &\rightarrow a \\ r < p &\rightarrow b \end{aligned} \tag{33}$$

La función p se encuentra en el rango de 0 a $\frac{1}{2}$, y sigue la forma de exponencial decreciente:

$$p = \frac{1}{2} e^{-K|f(a)-f(b)|} \tag{34}$$

El valor de K depende de la amplitud de valores que toma la función f y de un factor T que controla su pendiente.

$$K = \frac{T}{f_{max} - f_{min}} \tag{35}$$

Este factor T es el que decide la probabilidad de que sea elegida la muestra b en vez de a, además de la diferencia entre sus puntuaciones.

$$\begin{aligned} T \rightarrow 0 \Rightarrow p \rightarrow \frac{1}{2} \Rightarrow Pr(a) = Pr(b) = \frac{1}{2} \\ T \rightarrow \infty \Rightarrow p \rightarrow 0 \Rightarrow \begin{cases} Pr(a) = 1 \\ Pr(b) = 0 \end{cases} \end{aligned} \tag{36}$$

El primer caso es una elección aleatoria, donde no se tienen en cuenta las puntuaciones de los individuos; el segundo extremo se corresponde con el torneo simple, en el que se elige siempre la muestra con mayor puntuación.

Los torneos se asemejan al proceso de selección seguido en la Naturaleza por muchas especies, en el que dos individuos se miden para decidir cuál de ellos procreará.

Cada uno de estos métodos puede seguirse con o sin restitución de los elementos que se van seleccionando. La restitución permite a un individuo salir elegido varias veces, mientras que sin ella los elegidos una vez ya no entran en los sorteos posteriores. En la naturaleza, aquellos individuos que sobresalen entre sus congéneres tienen por regla general bastante más descendencia que el resto. En los algoritmos genéticos con restitución ocurre lo mismo; sin embargo, a veces es mejor limitar el número de veces que puede ser escogido un elemento, ya que si no condicionaría en gran medida las características de la siguiente generación, empobreciendo su diversidad genética y limitando mucho el área de estudio del problema que se quiere resolver, lo que puede derivar en el estancamiento de la población. En el otro extremo, si no se permite la restitución, se pierde la ventaja que tienen unos individuos sobre otros debido a su puntuación, facilitando demasiado que aquellos con baja puntuación se seleccionen, perdiendo el sentido del algoritmo. Si se permite la restitución sin límites la búsqueda de la solución se localiza demasiado; por el contrario, si no se permite, el algoritmo no avanza, la búsqueda no seguirá ninguna dirección. Por tanto hay que llegar a un compromiso, como que se permita la restitución hasta un número máximo de veces relacionado con el tamaño de la población, para asegurar que un individuo solo no condicione más de un porcentaje de la siguiente generación.

Existen otras variantes, como asegurar la selección del mejor especimen de la población o su paso directo a la siguiente generación para que no se pierda su información mientras no se encuentre otro mejor.

2.5. Nueva generación

Para crear un nuevo individuo hacen falta dos progenitores, cuyos códigos genéticos se combinarán para formar uno nuevo. El método de combinación más sencillo consiste en tomar un gen al azar como punto de corte; el nuevo individuo tendrá los genes del primero hasta el de corte iguales a los de uno de los progenitores y el resto como su otro progenitor. Siguiendo este sistema se pueden obtener dos descendientes distintos, como se muestra en la ilustración:

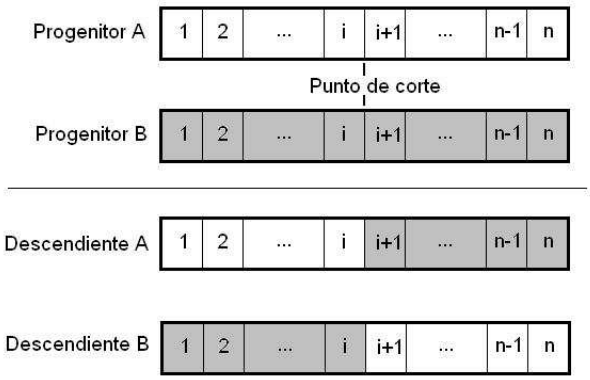

Figura 19. Transmisión genética con un punto de corte

En el ejemplo, el punto de corte se encuentra en el gen *i*. Nótese que nunca puede darse el caso de que el primer y el último gen de un hijo sean tomados del mismo progenitor; para evitar esta singularidad se añade otro punto de corte:

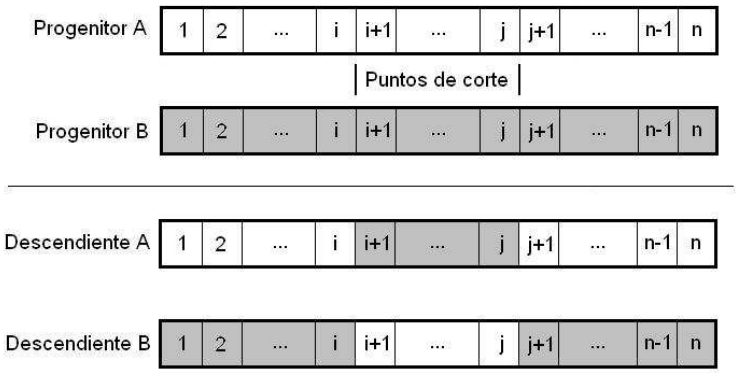

Figura 20. Transmisión genética con dos puntos de corte

Las **mutaciones** se introducen en este punto. La probabilidad de sufrir una mutación es la misma para todos los genes. Cuando un gen resulta mutado modifica el valor de su contenido. En el caso de genes binarios, el resultado de mutar el gen

b_i es $\bar{b}_i$[2]. El efecto de mutar un bit es mayor cuanto más peso tenga el bit. A modo de ejemplo, supongamos que tenemos una variable codificada con 6 bits, y que el valor de dicha variable en un individuo de la población es 33 (100001). Si mutamos el bit más significativo, el resultado será 1 (**0**00001); mientras que si la mutación recae sobre el 4º bit tendremos 37 (100**1**01). La variación en el primer caso es de 32 unidades, mientras que en el segundo sólo hay una distancia de 4. Para el caso de codificaciones en otras bases la mutación de un gen tiene varios posibles valores que tomar. Así, si tenemos una codificación genética en base 3, la variable anterior cuyo valor era 33, se expresará como (001020). Si mutamos el 4º gen, hay dos posibles valores que puede tomar: 42 (001**1**20) y 51 (001**2**20). Esto es equivalente a mutar varios genes binarios, por lo que la tasa de mutaciones debe descender conforme aumentamos el alfabeto genético.

[2]Operador negación: si $b = 1 \rightarrow \bar{b} = 0$ y viceversa

2.6. Funciones de Evaluación

A partir de las especificaciones del filtro que queremos conseguir, se puede representar una respuesta ideal:

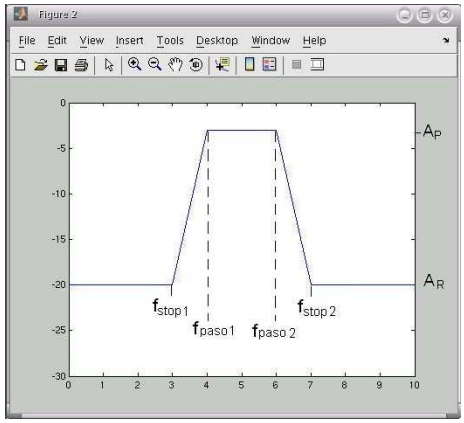

Figura 21. Filtro ideal

Al evaluar la respuesta de un filtro hay que tener en cuenta varios aspectos, como su comportamiento en las bandas de paso y de rechazo, y el tránsito de una a otra. A modo de ejemplo, en la figura siguiente se muestran dos respuestas diferentes:

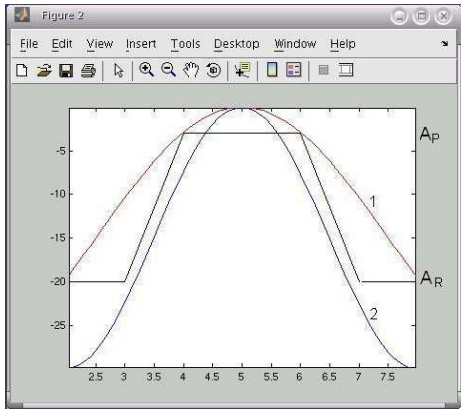

Figura 22. Comparación de 2 filtros reales y el ideal

La primera presenta mejor respuesta en la banda de paso, porque está por encima de la mínima atenuación para esta banda; sin embargo, el lento descenso hacia la franja de rechazo hace que no cumpla las especificaciones de rechazo en las inmediaciones de la banda de paso, alejándose de la respuesta óptima. El segundo filtro no tiene tan buena respuesta para la banda de paso, porque en los extremos de esta queda por debajo de las especificaciones; en cambio, tiene muy buen comportamiento en la región de rechazo. ¿Cuál de los dos filtros anteriores es mejor? ¿Cuánto? Como vemos, son varios los aspectos a tener en cuenta, además del problema de cuantificar la bondad de un filtro.

Para evaluar los filtros he analizado por separado sus respuestas en la banda de paso y en la de rechazo, para después promediar ambos resultados en una sola puntuación. Existen distintos criterios para realizar estas evaluaciones. El más sencillo consiste en contar el número de muestras de la respuesta del filtro que cumplen

las especificaciones iniciales: cuántas muestras de la banda de rechazo están por debajo de Ar y cuántas de la banda de paso están por encima de Ap. La figura siguiente muestra este método:

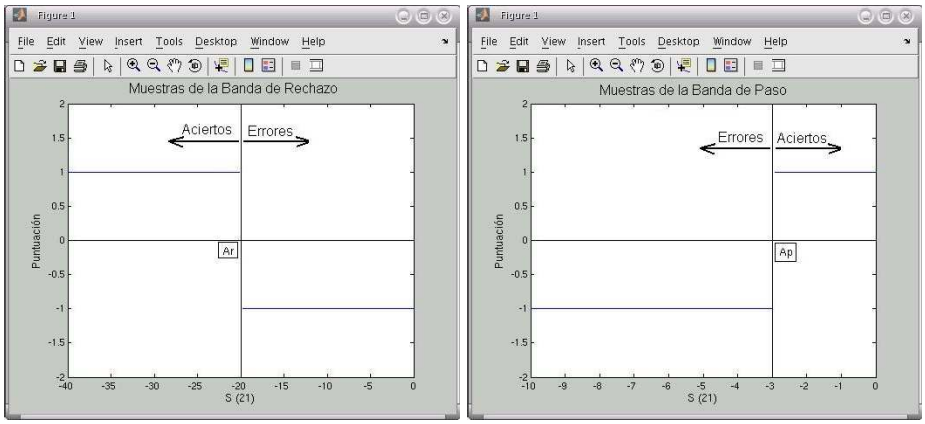

Figura 23. Funciones de evaluación de tipo 1: constantes

Las muestras que no cumplen las condiciones se restan a las anteriores, para resaltar los errores. La puntuación para la banda de paso P_{BP} es:

$$P_{BP} = \frac{\text{Nº de aciertos - Nº de errores}}{\text{Nº de muestras en Banda de Paso}} \tag{37}$$

En el caso de tener más muestras erróneas que válidas, el resultado sería negativo. Entonces hay que truncar el valor a 0, para no tener después problemas con los algoritmos de selección genética. La puntuación referente a la banda de rechazo P_{BR} se obtiene de manera análoga. Una vez tenemos ambas, calculamos la puntuación final del filtro:

$$P = \frac{P_{BP} + P_{BR}}{2} \tag{38}$$

El valor de las puntuaciones así obtenidas está comprendido entre 0 y 1. A pesar de su sencillez, este método no es muy apropiado, porque no distingue entre muestras con valores válidos pero distintos. Aunque sea mejor un filtro con más atenuación que otro en la banda de rechazo, el método los valorará como iguales si ambas atenuaciones son mayores que Ar; lo mismo sucede para la banda de paso. Para superar esta deficiencia, hay que utilizar evaluadores que asignen a cada muestra un valor según una función monótona continua que puede ser lineal o no. Estas funciones otorgan un valor comprendido entre 0 y 1 para las muestras válidas, y entre 0 y -1 para las no válidas.

Sean dos muestras a y b correspondientes a dos frecuencias de la misma banda del filtro. La puntuación que obtengan estará en relación con su valor, siendo iguales sólo si presentan la misma atenuación.

$$\begin{aligned} a, b \,\epsilon\, BP,\ At(a) > At(b) \rightarrow p(a) > p(b) \\ a, b \,\epsilon\, BR,\ At(a) < At(b) \rightarrow p(a) > p(b) \end{aligned} \tag{39}$$

Así se premia a las mejores muestras sobre las, aunque también válidas, no tan buenas, y se penaliza a las peores, permitiendo cuantificar la bondad de los filtros con un abanico de puntuaciones mucho más rico (continuo, y no discreto como antes). Las gráficas siguientes muestran el primer par de funciones de esta familia que utilicé:

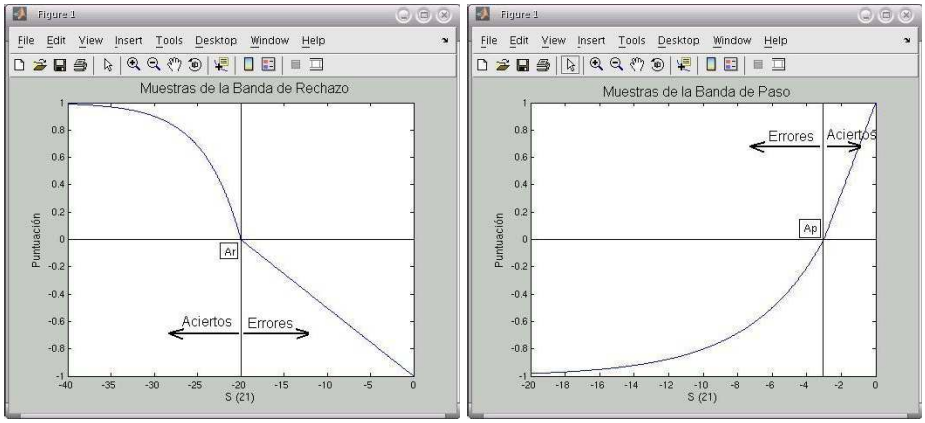

Figura 24. Funciones de evaluación de tipo 2: exponencial-lineal

Se trata de funciones exponenciales para el intervalo de valores $(-\infty, A)$ -siendo A un valor negativo cualquiera- y lineales para la región restante $(A, 0)$. La función exponencial para la banda de rechazo es de la forma:

$$P_{BR}^{pos} = 1 - e^{a(x-b)} \tag{40}$$

Donde a y b son los valores que determinarán, respectivamente, la pendiente y el origen de la función. Esta función satisface la condición de crecer hasta 1 conforme aumentamos la atenuación:

$$\lim_{x \to -\infty} P_{BR}^{pos}(x) = 1 \tag{41}$$

Como queremos que valga 0 (mínima puntuación) para atenuaciones iguales a la mínima especificada para la banda de rechazo del filtro (Ar), podemos deducir el valor de la constante b:

$$P_{BR}^{pos}(Ar) = 1 - e^{a(Ar-b)} = 0 \rightarrow b = Ar \tag{42}$$

Una vez fijados los extremos, hay que determinar la pendiente de la curva. Se ha tomado como criterio el que a una muestra que presente el doble de la atenuación mínima permitida ($Ar - 3\ dB$), le corresponda la puntuación de $0'5$:

$$P_{BR}^{pos}(Ar-3) = 1 - e^{a(Ar-3-b)} = \frac{1}{2} \rightarrow a = \frac{-\ln\left(\frac{1}{2}\right)}{3} \tag{43}$$

Con estas condiciones, ya queda definida la curva:

$$P_{BR}^{pos}(x) = 1 - e^{\frac{-\ln\left(\frac{1}{2}\right)}{3}(x-Ar)} \tag{44}$$

El parámetro a de la ecuación anterior determina la pendiente de la curva. Esta pendiente se hace más evidente en la proximidad del punto mínimo (Ar, donde la función vale 0). Cuanto mayor sea, más diferencia de puntuación habrá entre muestras cercanas a la atenuación mínima; y aumentar el gradiente en esta zona facilita al algoritmo evolutivo encontrar soluciones cada vez más alejadas del mínimo. Pero, si por el contrario, aumentamos mucho este parámetro, el resultado será una función prácticamente plana, como la utilizada en el primer caso, con los problemas asociados a ese tipo de funciones.

La sección recta del otro intervalo es:

$$P_{BR}^{neg}(x) = -1 + \frac{x}{Ar} \tag{45}$$

Esta recta cumple las condiciones impuestas para los extremos de su intervalo de actuación:

$$\begin{aligned} P_{BR}^{neg}(0) &= -1 \\ P_{BR}^{neg}(Ar) &= 0 \end{aligned} \tag{46}$$

De manera similar se calculan los parámetros de las funciones correspondientes a las frecuencias del ancho de banda:

$$\begin{aligned} P_{BP}^{neg}(x) &= -1 + e^{\frac{-\ln(\frac{1}{2})}{3}(x-Ap)} \\ P_{BP}^{pos}(x) &= 1 - \frac{x}{Ap} \end{aligned} \tag{47}$$

Para eliminar los tramos rectos en las puntuaciones, se ha hecho uso de otro tipo de funciones: las **sigmoides**. Una función sigmoidal es de la forma:

$$f(x) = \frac{1 - e^x}{1 + e^x} \tag{48}$$

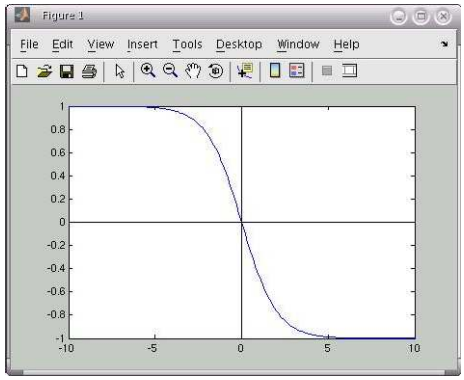

Figura 25. Función Sigmoidal

El centro de la curva y su pendiente se pueden modificar sustituyendo la variable x por una función lineal de x:

$$\begin{aligned} f(x) &= \frac{1 - e^{g(x)}}{1 + e^{g(x)}} \\ g(x) &= a(x - b) \end{aligned} \tag{49}$$

De esta forma conseguimos desplazar el punto 0 hasta b y cambiar la velocidad de convergencia con a. Utilizando estas funciones, se construye el siguiente evaluador:

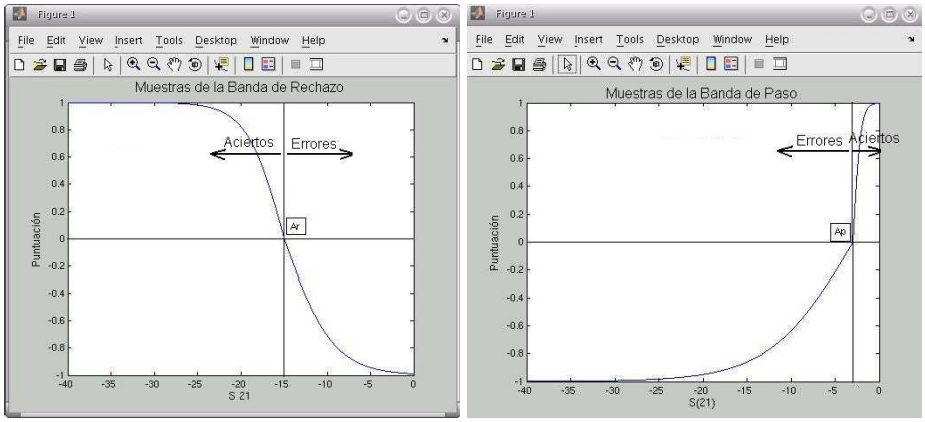

Figura 26. Funciones de evaluación de tipo 3: sigmoidales

Cada tramo corresponde a una función sigmoidal distinta, ajustando su origen y gradiente para cada caso. Veamos cómo construir la curva correspondiente a las muestras de frecuencias de la banda de rechazo que presentan una atenuación mayor a Ar:

$$\begin{aligned} P_{BR}^{pos}(x) &= \frac{1 - e^{g(x)}}{1 + e^{g(x)}} \\ g(x) &= \frac{-7}{Ar}(x - Ar) \end{aligned} \tag{50}$$

En este caso, tenemos:

$$\begin{aligned} a &= \frac{-7}{Ar} \\ b &= Ar \end{aligned} \tag{51}$$

La curva está centrada en Ar y la pendiente es inversamente proporcional a este valor, porque si no se saturaría enseguida. El valor de 7 se ajustó manualmente tras realizar diversas pruebas.

La puntuación para las muestras de esta banda que no cumplen la condición de atenuación sigue la curva:

$$P_{BR}^{neg}(x) = \frac{1 - e^{a(x-Ar)}}{1 + e^{a(x-Ar)}} \tag{52}$$

El valor de a se obtiene de manera análoga a como se hizo anteriormente para fijar la pendiente de una curva exponencial:

$$\begin{aligned} p(x) &= \frac{1 - e^{a(x-b)}}{1 + e^{a(x-b)}} \\ e^{a(x-b)} = \frac{1 - p(x)}{1 + p(x)} \rightarrow a(x - b) &= \ln\left(\frac{1 - p(x)}{1 + p(x)}\right) \\ a &= \frac{\ln\left(\frac{1-p(x)}{1+p(x)}\right)}{x - b} \end{aligned} \tag{53}$$

En este caso se ha determinado que una muestra de la banda de rechazo que presente una atenuación de Ap (la atenuación máxima para la banda de paso) debe obtener una puntuación muy baja: $-0'98$. Con esta condición, tenemos:

$$\begin{aligned} p(Ap) - 0'98 \quad b &= Ar \\ a &= \frac{\ln\left(\frac{1+0'98}{1-0'98}\right)}{Ap - Ar} \end{aligned} \tag{54}$$

Esta curva presenta una buena velocidad de saturación para este intervalo. Nótese la diferencia con la recta del modelo anterior, que otorga penalizaciones menores. El efecto conseguido con la sigmoide en este intervalo es similar al mencionado para el tramo anterior, mejorando los resultados obtenidos por los algoritmos de optimización.

Las funciones utilizadas para evaluar las muestras de la banda de paso son:

$$P_{BP}^{pos}(x) = \frac{1 - e^{\frac{7}{Ap}(x-Ap)}}{1 + e^{\frac{7}{Ap}(x-Ap)}} \tag{55}$$

$$P_{BP}^{neg}(x) = \frac{1 - e^{a(x-Ap)}}{1 + e^{a(x-Ap)}} \tag{56}$$

$$a = \frac{\ln\left(\frac{1+0'95}{1-0'95}\right)}{Ap - Ar} \tag{57}$$

En este caso se ha tomado como punto para fijar la curva la puntuación de $-0'95$ para muestras del ancho de banda con atenuación Ar. No se tomado una pendiente mayor porque durante el desarrollo de las iteraciones genéticas aparecerán muchos individuos con el ancho de banda desviado, y es preferible mantener mayor intervalo de atenuaciones sin saturar, a fin de tener una pendiente clara que seguir en la evolución.

3. Optimización por Gradiente

3.1. Introducción

Las funciones matemáticas se utilizan para modelar sistemas del mundo real (bien sea de la Naturaleza, evoluciones económicas, etc). El análisis de dichas funciones es una rama importante de las matemáticas, ya que permite conocer mejor los procesos modelados: predecir su comportamiento, compararlo con otros sistemas, mejorar un diseño. Entre las cualidades a analizar en una función están su derivabilidad, máximos y mínimos, raíces o continuidad. En nuestro caso queremos estudiar la función de evaluación de los filtros para optimizar el diseño de los mismos. Para ello vamos a hacer uso de una serie de técnicas que se han desarrollado para el análisis de funciones de error y de coste. Las primeras se utilizan en un proceso de aprendizaje clasificativo, asignando a cada muestra de una población un valor proporcional al acierto de su clasificación; las funciones de coste relacionan el coste asociado a una determinada decisión para cada muestra, y se usan en la evaluación de distintas decisiones en un proceso de diseño. El objeto de estas funciones es encontrar un conjunto de parámetros que las minimice: cuanto menor es el coste asociado a una decisión, más rentable es esta; de igual manera, cuantos menos errores se cometen o menores son estos, mejor es el aprendizaje. Por tanto, se analizan las funciones y se trata de minimizarlas mediante diversas técnicas. El punto que nos proporcione su mínimo será el punto óptimo que buscamos.

Queremos maximizar la función de evaluación de los filtros en el espacio de las variables de simulación. Como estos algoritmos son de minimización, lo que hacemos es transformar la función restándola a una constante K. De esta forma, los máximos pasan a ser mínimos y viceversa:

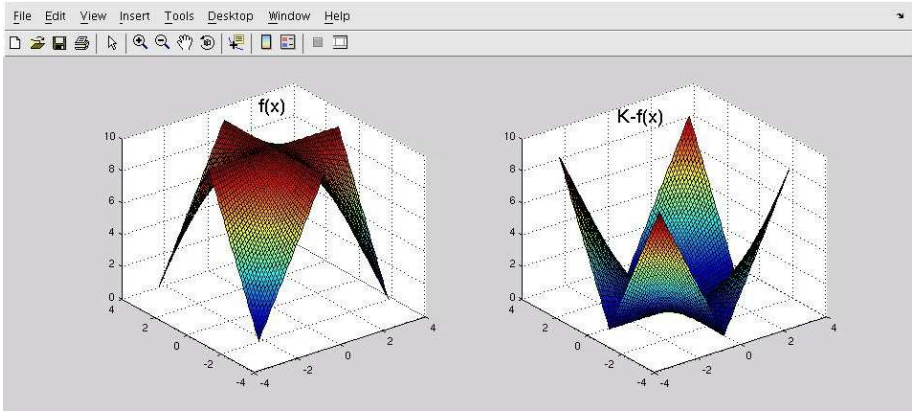

Figura 27. Maximizar una función es minimizar la función opuesta

El objetivo consiste en encontrar un vector de pesos $\vec{w}$ que minimice la superficie de error (para los cuales la superficie de error sea mínima) [1]. *E(w)* es la función de error sobre los pesos. Esta función representa una superficie en el espacio de los pesos: $E(w_0, w_1, w_2, ..)$. Hay que encontrar un vector wm tal que $E(wm_0, wm_1, wm_2, ...) = min(E(w))$.

La función de error para nuestro caso es no lineal, por lo que no puede analizarse con los métodos convencionales. Además, no se dispone de un modelo a seguir de dicha función. Lo que conocemos de ella es su valor para cualquier vector de pesos que analicemos, calculado este mediante el análisis de la respuesta de un

individuo en el simulador **FEST** y, posteriormente, evaluado con **MATLAB**.

El gradiente de una función en un mínimo es cero, al igual que sucede en los máximos y en los puntos de silla (máximos por un extremo y mínimos por el otro):

$$\nabla E = 0 \tag{58}$$

Además, existen distintos mínimos en una función de este tipo: los mínimos locales y el mínimo global. Los primeros sólo son mínimos en una determinada región de la función, mientras que el mínimo global lo es para toda la función. Vemos por tanto, que el gradiente de una función es una herramienta útil para encontrar su mínimo, pero no es determinante; hay que combinarla con otros métodos.

Para encontrar el mínimo global de una función no lineal como la nuestra, tenemos que utilizar algoritmos iterativos en los cuales, partiendo de un punto, nos vamos acercando progresivamente a la solución. Para cada iteración, el nuevo punto donde se evalúa la función se obtiene a partir del anterior, sumándole un incremento:

$$\vec{w}_{(i+1)} = \vec{w}_{(i)} + \Delta\vec{w}_{(i)} \tag{59}$$

Existen distintos tipos de algoritmos con esta filosofía, que difieren entre sí en la forma de calcular el incremento de actualización de pesos. La elección del peso inicial determina hacia qué mínimo se dirigirá el algoritmo, pudiendo converger hacia uno local. La presencia de puntos de silla o de tramos con poca pendiente en la región de búsqueda puede ralentizar el proceso de búsqueda. Muchos de estos algoritmos responden bien ante funciones sencillas o monótonas, como una superficie cuadrática, pero tienen problemas con funciones más complejas; por ello se debe elegir bien el punto de partida, reduciendo la región de búsqueda desde el principio.

Todos los algoritmos comienzan inicializando los pesos a un valor aleatorio, que determinará su evolución, incluso para aquellos capaces de escapar de mínimos locales, como el de descenso por gradiente.

Existen varios criterios para detener el proceso iterativo:

- Después de un número determinado de iteraciones. Parece una elección bastante lógica, pero a veces es difícil determinar cuántas iteraciones son necesarias. Con pocas iteraciones, el algoritmo será más rápido, pero puede que el resultado no sea muy bueno si necesitase más iteraciones para converger. También puede suceder que hagamos iteraciones innecesarias cuando ya ha convergido.
- Después de un tiempo determinado. El tiempo que tarda en realizar una optimización suele ser directamente proporcional al número de iteraciones corridas, por lo que este criterio, aunque práctico, presenta el mismo problema del caso anterior: determinar el tiempo idóneo.
- Después de alcanzar o rebasar un valor determinado. Supongamos que necesitamos encontrar una solución que proporcione un errorr menor que un valor prefijado. El algoritmo no se detendrá hasta alcanzar ese punto. La dificultad consiste en elegir dicho valor. Si el valor elegido no puede ser alcanzado, el algoritmo entraría en un bucle infinito.
- Cuando el incremento entre iteraciones es muy pequeño. Cuando el algoritmo se encuentra en un punto próximo al mínimo, los avances que realiza son cada vez menores. Llega un momento en el que la mejora entre iteraciones es tan pequeña que no merece la pena seguir iterando, por lo que detenemos el algoritmo. Corremos el riesgo de terminar prematuramente si se entra en una zona de baja pendiente lejos del mínimo.

- Cuando la función empieza a incrementarse. Si el resultado de la iteración $n+1$ es mayor que el de la iteración n, la mayoría de las veces el mínimo estará en un punto intermedio (aunque indeterminado). Hay que tener presente que no siempre sucede así.

Dado que todas las opciones tienen sus puntos a favor y en contra, la mejor estrategia consite en combinar varios de estos criterios para asegurar un correcto funcionamiento.

3.2. Descenso por Gradiente

Es el algoritmo de minimización iterativo más simple. Se comienza por un peso al azar y se va actualizando en la dirección del gradiente negativo en cada nuevo punto:

$$\Delta w_i = -\rho \nabla f(w_i) \tag{60}$$

El valor de la función irá progresivamente descendiendo hasta encontrar un mínimo, en el cual el gradiente es cero y el algoritmo se detiene.

En la ecuación 60, ρ es la tasa de aprendizaje y su valor puede ser constante o irse actualizando en función del resultado. También puede ir decreciendo continuamente, aunque esto puede ocasionar que la convergencia sea muy lenta. En general, el uso de una tasa constante es una buena elección, aunque entonces no se garantiza la convergencia. Si la tasa ρ es muy grande, el algoritmo puede terminar oscilando sobre la solución; si es muy pequeña, tardará demasiado en llegar a la solución.

En muchas superficies el gradiente en un punto no apunta directamente al mínimo, con lo que este algoritmo recorrerá un camino oscilatorio muy largo en su aproximación. Esto sucede en superficies con valores propios dispares, como la de la figura:

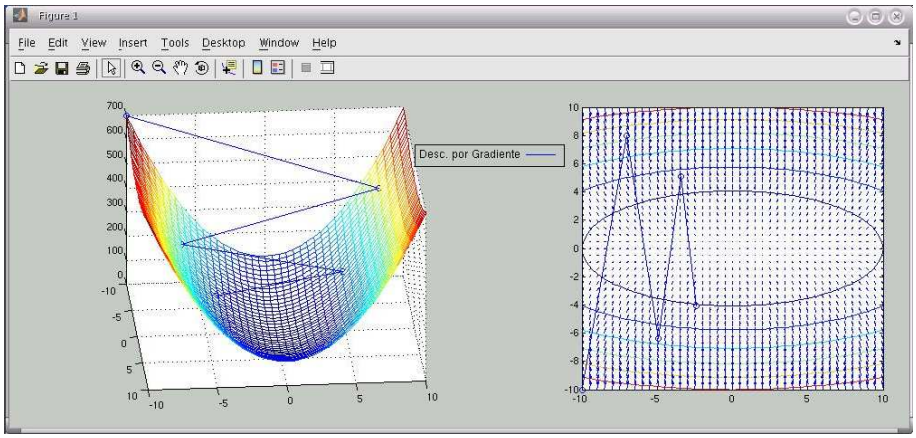

Figura 28. Descenso por gradiente

En este caso se ha elegido detener el algoritmo después de 4 iteraciones, quedando bastante lejos del mínimo. Debido a las constantes oscilaciones que realiza en su evolución, necesitaría algunas iteraciones más para acercarse visiblemente más al mínimo de la función.

3.3. Descenso por Gradiente con Momento

Para evitar este problema de oscilaciones, añadimos un término al incremento de pesos. Este término es proporcional al valor del peso anterior, por lo que se añade memoria al algoritmo:

$$\Delta w_{i)} = -\rho \nabla f(w_i) + \mu \Delta w_{i-1} \tag{61}$$

Esto implica cierta inercia; en superficies con poca pendiente, los incrementos se irán haciendo mayores, acelerando el resultado. Por el contrario, en situaciones como la de la figura anterior, en la que los sucesivos pasos oscilan como un diente de sierra, se disminuye dicha oscilación. Esto es debido a que la componente perpendicular al avance en un paso es de sentido contrario a la siguiente, por lo que al sumar el término del momento tenderán a anularse. La figura compara ambos descensos por gradiente, con el término de momento y sin él:

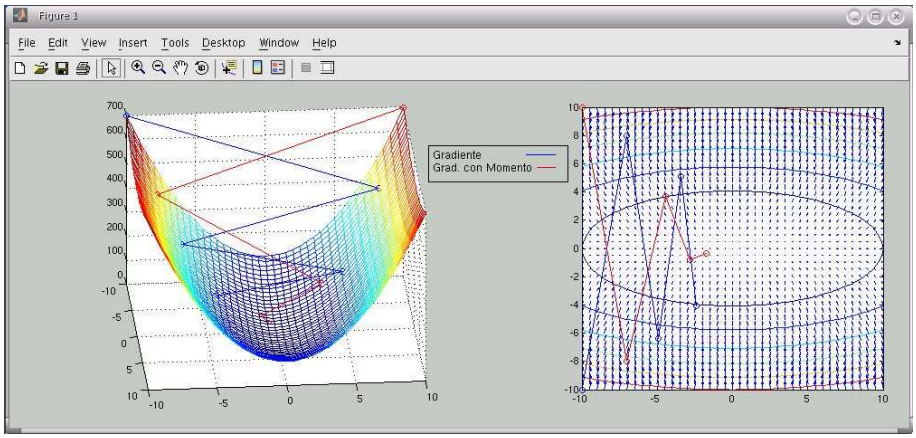

Figura 29. Descenso por gradiente con momento

Vemos cómo la oscilación se reduce, lo que permite una optimización más directa. La aproximación que consigue en 4 iteraciones es mejor que utilizando el descenso sin momento, aunque todavía no es muy eficiente.

3.4. Búsqueda en línea

Primero debemos decidir la dirección en la que movernos y luego cuánto movernos. Con el descenso por gradiente, la dirección viene dada por el negativo del gradiente de la función en cada punto, mientras que la longitud del avance se determina a priori, aleatoriamente. Sería mejor desplazarnos en la dirección elegida hasta que el gradiente sobre dicha dirección se minimice. Así evitaríamos dar sucesivos pasos en la misma dirección. La búsqueda en línea representa la minimización de un problema unidimensional en un solo paso.

$$\begin{aligned} w_{i+1} &= w_i + \lambda_i d_i \\ \lambda_i &\rightarrow min \nabla f(w_i + \lambda_i d_i) \end{aligned} \tag{62}$$

El parámetro λ controla la distancia que hay que desplazarse en la dirección escogida. Para evitar tener que calcular el gradiente en la dirección escogida, se pueden tomar sucesivas muestras sobre esta dirección y quedarnos con el último punto antes de que el gradiente comience a crecer; aunque esto no determina necesariamente el mínimo, sirve como aproximación. El mínimo puede entonces determinarse por interpolación parabólica.

3.5. Gradiente Conjugado

Como hemos visto, el uso de sucesivos vectores de gradiente no es la mejor opción para encontrar el mínimo. Si escogemos λ tal que se minimice la función sobre la dirección d, tenemos:

$$\frac{\partial}{\partial \lambda} f(w_i + \lambda d_i) = 0 \tag{63}$$

Esto indica que el gradiente en el punto w_{i+1} es perpendicular a la dirección d_i, por lo que la siguiente dirección que se tome, d_{i+1}, también lo será.

$$\nabla f(w_{i+1})^T d_i = 0 \tag{64}$$

Podemos tener, por tanto, el problema de que el algoritmo oscile continuamente en su búsqueda del mínimo. Para evitarlo, introducimos el concepto de direcciones conjugadas. Dos direcciones son conjugadas si la componente del gradiente sobre la primera de ellas es mínima en el punto de intersección de ambas. Nótese que si esta componente no es nula, las direcciones no son perpendiculares. Esta condición puede expresarse de la siguiente manera:

$$d_{i+1}^T H d_i = 0 \tag{65}$$

H es el Hessiano evaluado sobre el punto w_{i+1}. El problema aquí consiste en calcular la matriz H, ya que se trata de una información de segundo orden sobre la función y a menudo se desconoce. En muchos casos se recurre al uso de aproximaciones para agilizar el proceso.

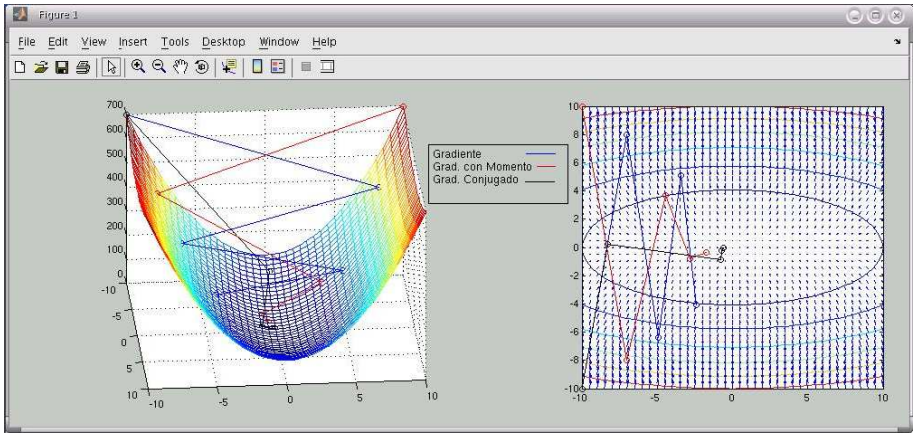

Figura 30. Gradiente conjugado

En la representación se muestran los tres métodos estudiados. Puede observarse la mejora que supone este algoritmo respecto de los anteriores. Al tratarse de una superficie cuadrática, a pesar de la disparidad de los valores propios de su elipse, en la 2^a iteración ya se encuentra en las inmediaciones del mínimo.

3.6. Método de Newton

Partiendo de la expansión de Taylor para una función:

$$f(x) = f(x_0) + (x - x_0)\nabla f(x_0) + \frac{1}{2}(x - x_0)H(x - x_0) \tag{66}$$

Esta expresión es válida para puntos cercanos a x_0. Supongamos que en este punto hay un mínimo, con lo que quedaría de la siguiente forma:

$$f(x) = f(x_0) + \frac{1}{2}(x - x_0)H(x - x_0) \tag{67}$$

El método de Newton consiste en realizar una aproximación cuadrática alrededor del mínimo. En una superficie cuadrática el Hessiano es constante:

$$\begin{aligned} f &= k + \sum_{i=1}^{n} a_i xi + \sum_{i=1}^{n} b_i xi^2 \\ \nabla f &= (a_1 + 2b_1x_1, a_2 + 2b_2x_2, ..., a_n + 2b_nx_n) \\ H &= \begin{pmatrix} 2b_1 & 0 & ... & 0 \\ 0 & 2b_2 & ... & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & ... & 2b_n \end{pmatrix} \end{aligned} \tag{68}$$

La expansión de Taylor para puntos cercanos a un mínimo en una superficie cuadrática queda así:

$$f(x) = f(x_0) + \frac{1}{2}H(x - x_0)^2 \tag{69}$$

Si derivamos dicha ecuación obtenemos:

$$\nabla f(x) = H(x - x_0) \tag{70}$$

Ahora podemos determinar el mínimo de la función:

$$x_0 = x - H^{-1}\nabla f(x) \tag{71}$$

El método de Newton para hallar el mínimo parte de la hipótesis de que la aproximación cuadrática es válida en el entorno del mínimo, e implica el cálculo del inverso de la matriz Hessiana. La ecuación anterior nos indica el incremento que debemos tomar entre muestras:

$$w_{i+1} = w_i + \Delta w_i = w_i - H^{-1}\nabla f(w_i) \tag{72}$$

Este vector se conoce como dirección de Newton y en una superficie cuadrática apunta directamente al mínimo:

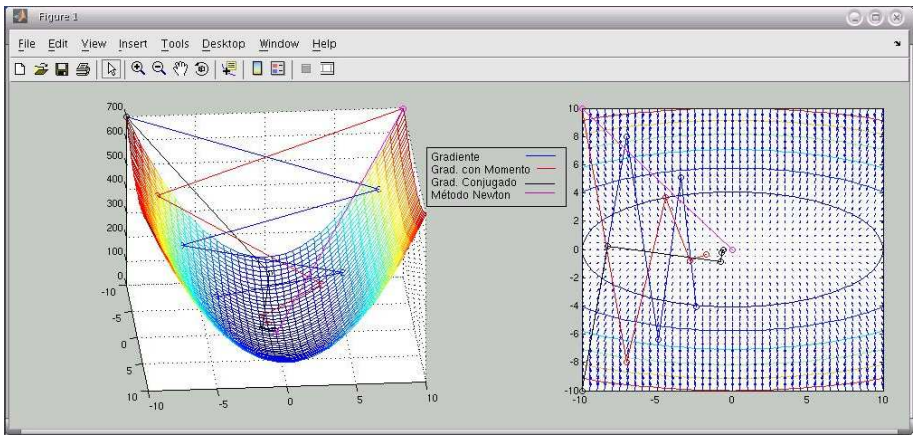

Figura 31. Método de Newton

El método presenta varias dificultades: el cálculo de la matriz Hessiana en funciones no lineales, y el hecho de que la dirección de Newton también puede apuntar a un máximo o a un punto de silla. Para evitar el primer problema se puede desarrollar una aproximación de la inversa de la matriz Hessiana utilizando sólo información de primer nivel sobre la función. Es lo que se conoce como el algoritmo de Quasi-Newton. En este algoritmo se asegura que el valor de la función en una iteración no es mayor que el de la anterior, como ocurre con el gradiente conjugado, pero puede estancarse en un mínimo local.

Existen muchos otros algoritmos de minimización; como el de Levenberg-Marquardt, diseñado para minimizar sumas cuadráticas de errores.

4. Interfaz Gráfica y resultados

A continuación se presentan una serie de ejemplos de diseño utilizando la herramienta desarrollada.

4.1. Filtro de Orden 1 a 14 GHz

Se ha desarrollado una aplicacion en **Matlab** para diseñar y optimizar filtros de microondas. Como ya vimos, el primer paso en el diseño de un filtro consiste en definir las especificaciones que debe cumplir, definir su banda de paso y su selectividad, tal como se muestra en el esquema siguiente:

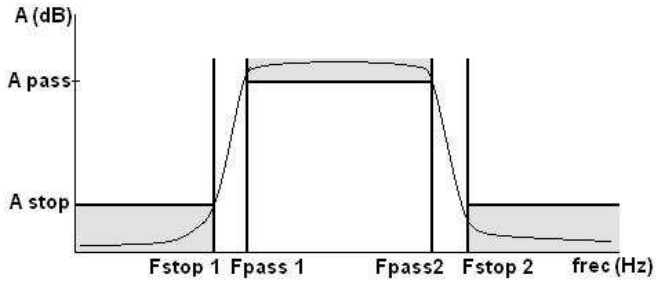

Figura 32. Parámetros de un filtro Paso Banda

Esto puede hacerse desde la ventana principal del programa:

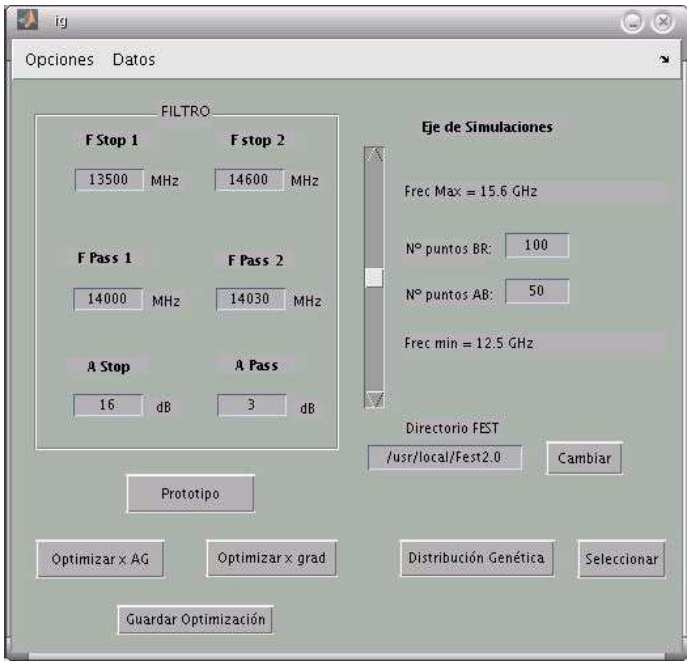

Figura 33. Ventana principal del programa

En este caso se ha querido diseñar un filtro paso banda centrado en 14 GHz con un ancho de banda de 30 MHz. Las frecuencias de corte fijan además el intervalo de frecuencias que se analizará en las simulaciones que se lleven a cabo. Dicho

intervalo se puede ampliar o reducir con la barra del centro. Una vez modificadas las especificaciones de acuerdo con el filtro que buscamos, el botón *Prototipo* permite implementar una primera aproximación, desarrollada mediante el algoritmo de Chebyshev, que nos indica el orden del filtro que necesitamos y proporciona unas dimensiones desde las que partir en la optimización que se realizará después. Para los valores que se han introducido a modo de ejemplo obtenemos un filtro de orden 1 cuya respuesta en el intervalo de frecuencias seleccionado (12.5 a 15.6 GHz) es:

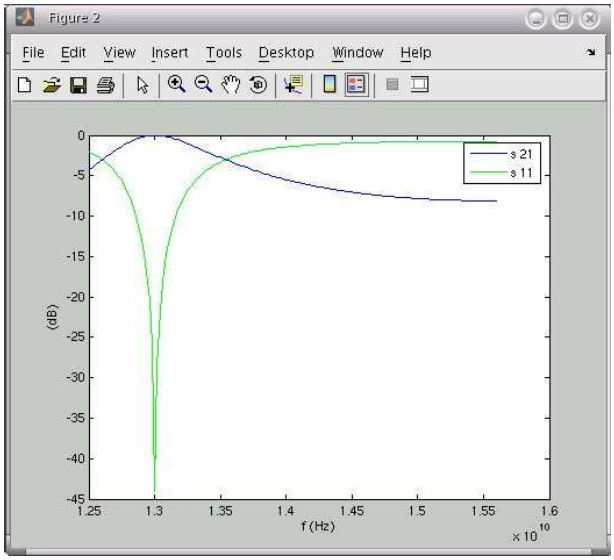

Figura 34. Respuesta en frecuencia del filtro prototipo

Recordemos que los parámetros S_{21} y S_{11} representan la energía transmitida y reflejada, respectivamente. Asímismo se muestra una imagen con las dimensiones, en milímetros, que tienen las distintas partes que forman este filtro:

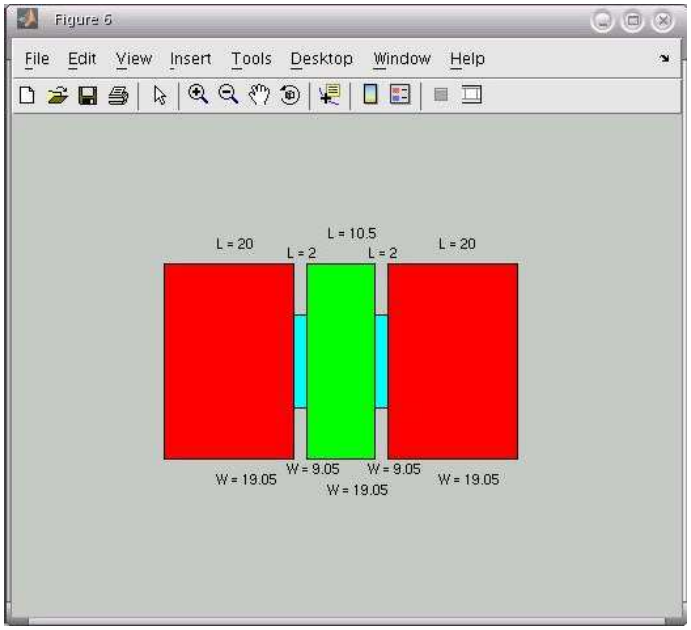

Figura 35. Dimensiones (mm) del filtro prototipo

Los rectángulos rojos representan los puertos de entrada y de salida del filtro; los rectángulos verdes son las cavidades resonantes y los azules represntan los escalones inductivos que separan una cavidad con otra o con un puerto. Esta representación se corresponde con la siguiente figura:

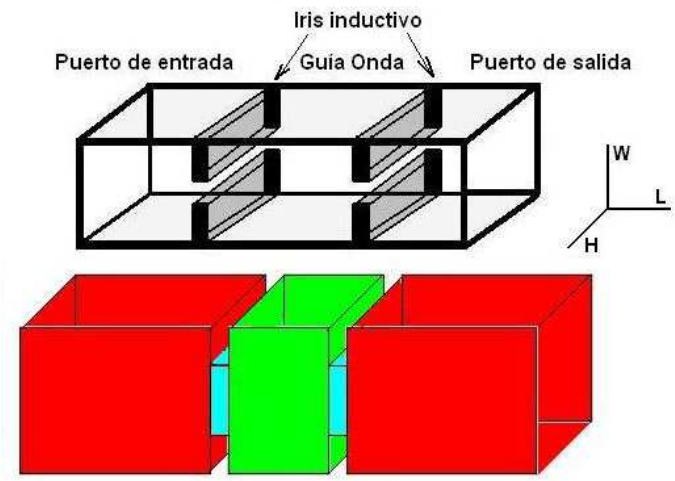

Figura 36. Representación tridimensional de un filtro de orden 1

Nótese que sólo se proporcionan las medidas de dos dimensiones; esto es debido a que se han elegido guías rectangulares cuya altura es constante $H = 9{,}525mm$, que es la mitad de la anchura de la guía $W = 19{,}05mm$.

Vemos que la respuesta del filtro obtenido por este sistema no satisface las especificaciones, ya que su banda de paso aparece sobre los 13 GHz en vez de los 14 GHz que se requieren. Para mejorar el diseño habrá que realizar una optimización mediante algoritmos genéticos. Hay que elegir qué variables se van a optimizar y de qué forma se va a hacer. Esto puede hacerse desde la pestaña *Opciones* que aparece en la esquina superior izquierda.

- La primera opción muestra los **parámetros del algoritmo genético**: desde aquí se controla el tamaño de la población, el número de bits por variable que se usarán, la base del alfabeto de estas variables, la probabilidad de mutación, el número de iteraciones que se llevarán a cabo, el número de ordenadores que realizarán las simulaciones en paralelo[3], y el proceso de selección de individuos que se seguirá: ruleta, torneo simple o torneo probabilístico.

[3]Ver Anexo: Procesado en paralelo

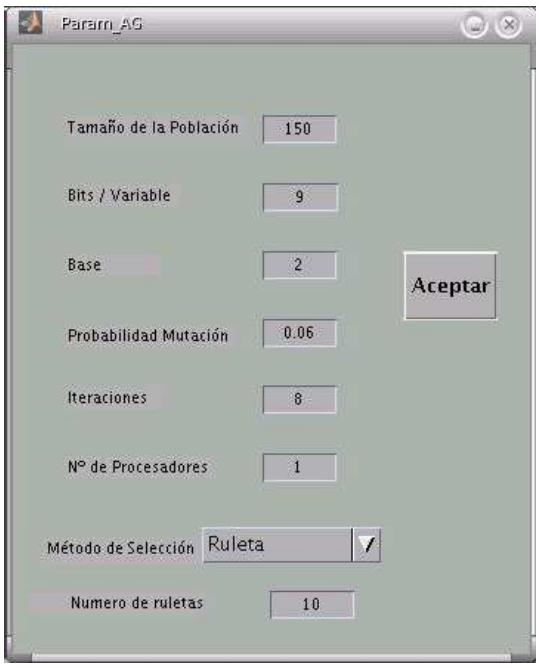

Figura 37. Ventana de Parámetros del A.G.

- La segunda opción es **Variables de Simulación**: desde aquí se pueden seleccionar las dimensiones de la estructura que se quieran mejorar. Estas dimensiones son las anchuras W y longitudes L tanto de las cavidades resonantes como de los escalones. Dependiendo de su orden, el filtro tendrá un número determinado de estos elementos, que serán los únicos que se muestren habilitados. En este caso, disponemos de una cavidad y dos discontinuidades inductivas, por lo que sólo podemos seleccionar la primera cavidad y los escalones 1 y 2. La opción *All* permite tratar todos los elementos de una misma clase por igual; si, por ejemplo, seleccionamos esta opción para las longitudes de los escalones, se asignará una única variable para todas las longitudes de escalones que tiene el filtro. Esto es distinto de seleccionar individualmente todos los escalones, ya que entonces tendremos una variable por cada escalón, pudiendo resultar en longitudes diferentes para cada uno de ellos. Por último, el botón *Simétrico* hace que las dimensiones de la estructura optimizada sean simétricas respecto del eje central del filtro. Una vez seleccionadas las variables que deseamos, pulsamos el botón *Aceptar* y cerramos la ventana.

Figura 38. Ventana de Variables de simulación

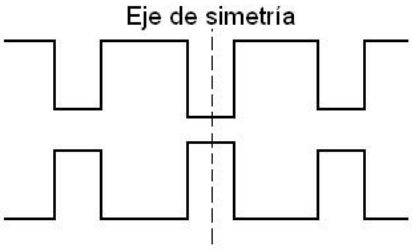

Figura 39. Filtro simétrico

Una vez realizada la configuración, volvemos a la ventana principal del programa y pulsamos el botón **Optimizar x AG**. La optimización comenzará automáticamente. Hay que asegurarse de que la ruta del programa *Fest* que aparece en la casilla inferior izquierda es la correcta; si no lo es, cámbiese. El tiempo que tarde el programa en realizar la optimización dependerá del número de iteraciones y del tamaño de la población. Al finalizar aparece el esquema del modelo optimizado y su respuesta en frecuencia, donde se puede comparar con el diseño inicial.

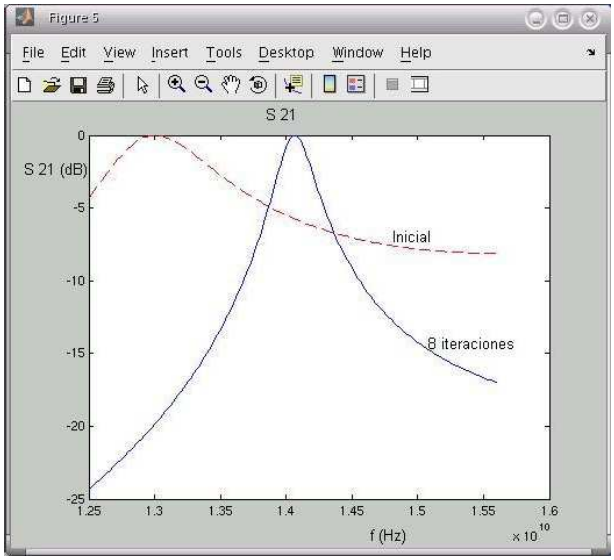

Figura 40. Comparación del filtro inicial y una primera optimización

Vemos como en este ejemplo se ha conseguido centrar la banda de paso en 14 GHz y aumentar su selectividad, tal como se quería, aunque el ancho de banda obtenido todavía es muy grande.

La elección de los parámetros del algoritmo genético seguido condiciona la evolución del mismo y, por tanto, también el resultado final. Para saber si la configuración escogida es apropiada para la optimización que queremos, podemos visualizar una serie de gráficas indicativas de la evolución seguida en las distintas iteraciones del algoritmo. La primera gráfica muestra el número de individuos de una población que superan el mejor resultado alcanzado hasta entonces, acercando más la respuesta real del filtro a la ideal. Esto permite ver si los cruces que se realizan son efectivos o no. En la misma imagen, en la gráfica inferior se representan la media, el máximo y el mínimo de las puntuaciones de cada población.

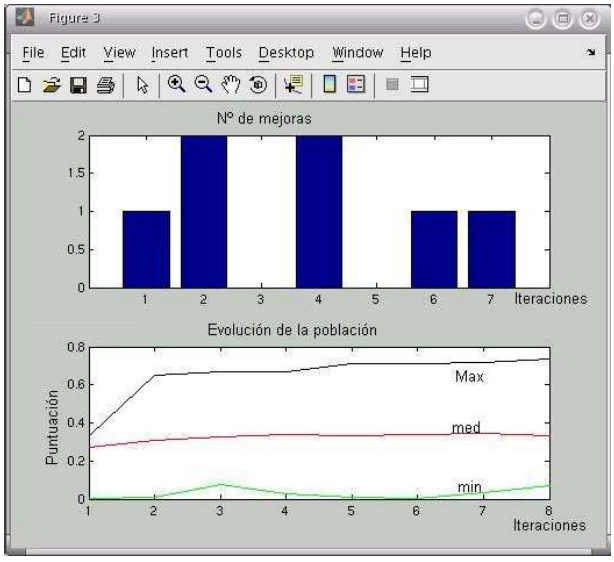

Figura 41. Evolución de la puntuación de la población en 8 iteraciones

La primera imagen nos indica que en casi todas las iteraciones se consiguen mejoras, lo cual es buena señal. En la segunda gráfica observamos que la mayoría de esas mejoras son pequeñas, ya que la curva de máximos tiene poca pendiente. También podemos deducir que la población no se ha focalizado en torno a un punto o región, sino que mantiene su área de exploración; en caso contrario, habría menos diferencia entre las tres curvas en las últimas iteraciones.

Pulsando el botón **Distribución Genética** se accede a una serie de gráficas que muestran la riqueza genética de la población en todas las iteraciones realizadas. En estas gráficas aparecen el número de individuos de la población que tienen un determinado bit igual, para todos los bits y todas las variables. Como para este caso se manejan 3 variables aparecen 3 gráficas, cada una con la distribución de sus 9 bits. En la primera población (iteración 1) tenemos una distribución uniforme, en la que no destaca ningún valor sobre el otro:

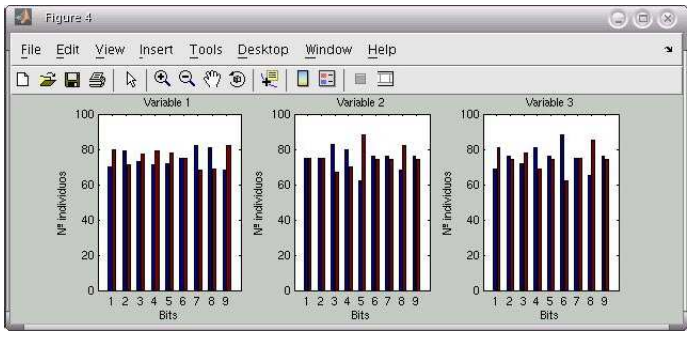

Figura 42. Distribución genética uniforme en la población inicial

Si avanzamos unas pocas iteraciones, empieza a haber cierta tendencia en algunos bits, sobre todo en la tercera variable:

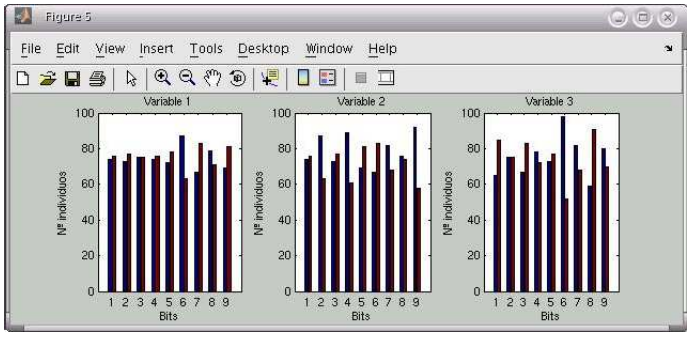

Figura 43. Distribución genética en la 4ª iteración

Esto es señal de que el algoritmo toma una dirección determinada para dicha variable, centrando la población en una región concreta. La última gráfica muestra la distribución de bits en la iteración final:

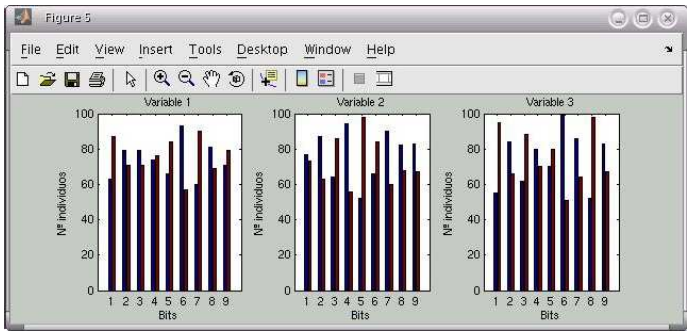

Figura 44. Distribución genética en la 8ª iteración

La población mantiene su riqueza genética, tal como habíamos predicho observando la evolución de los parámetros media, máximo y mínimo. Hay algunos bits que presentan un claro predominio de un valor sobre el otro; pero esta tendencia no está muy acentuada, y en posteriores iteraciones podría invertirse.

Dado que el proceso de optimización parece desarrollarse satisfactoriamente bajo las condiciones escogidas, ejecutaremos 8 iteraciones más con la misma configuración. Para ello basta con volver a pulsar el botón **Optimizar x AG**. La respuesta del nuevo filtro que conseguimos ajusta mejor el ancho de banda:

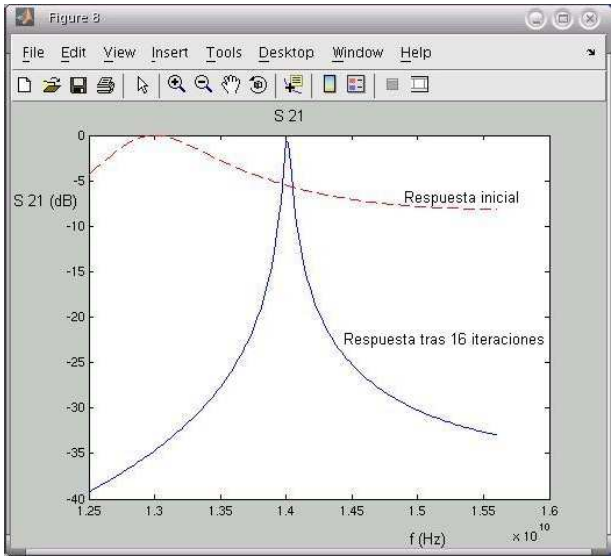

Figura 45. Respuesta del filtro tras 16 iteraciones

La evolución de resultados para estas 8 nuevas iteraciones pone de manifiesto dificultades para conseguir mejoras significativas durante las 6 primeras iteraciones; pero en el útlimo cruce se han logrado individuos notables. Esta incertidumbre es una característica propia de los algoritmos genéticos, relacionada con el azar implícito en su funcionamiento.

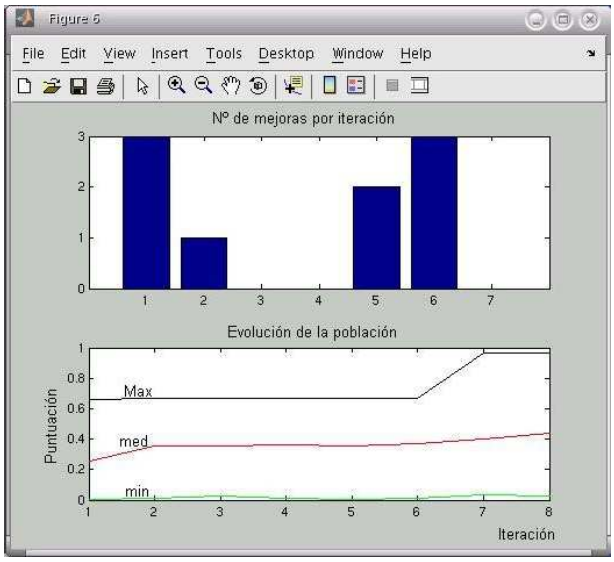

Figura 46. Evolución de la población entre las iteraciones 9 y 16

Veamos cómo está la distribución genética en la última iteración:

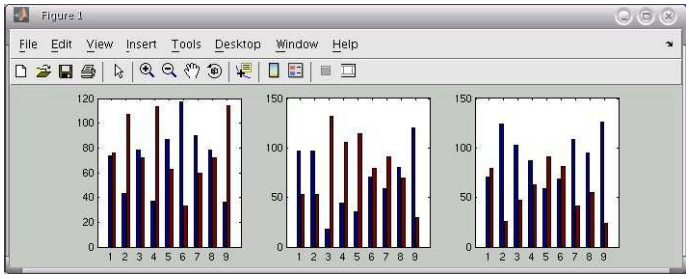

Figura 47. Distribución genética en la iteración 16

La mayor parte de los genes de la población presentan una clara preferencia por un valor. Esto indica que muchos individuos tendrán características similares entre sí, por lo que en posteriores cruces se heredarán estas características comunes: la población tiene poca riqueza genética. Si seguimos iterando sobre esta población, las probabilidades de mejorar el diseño actual serán cada vez menores. Es mejor cambiar de método, para lo cual tenemos dos opciones: optimización por gradiente o empezar el proceso partiendo de otro individuo (en lugar del prototipo teórico inicial). Estudiaremos ambas opciones.

La primera opción consite en realizar una optimización por gradiente a partir del mejor individuo de la población. Pulsamos el botón **Optimizar x Grad** y el proceso comenzará automáticamente. El resultado que se obtiene presenta la siguiente respuesta en frecuencia:

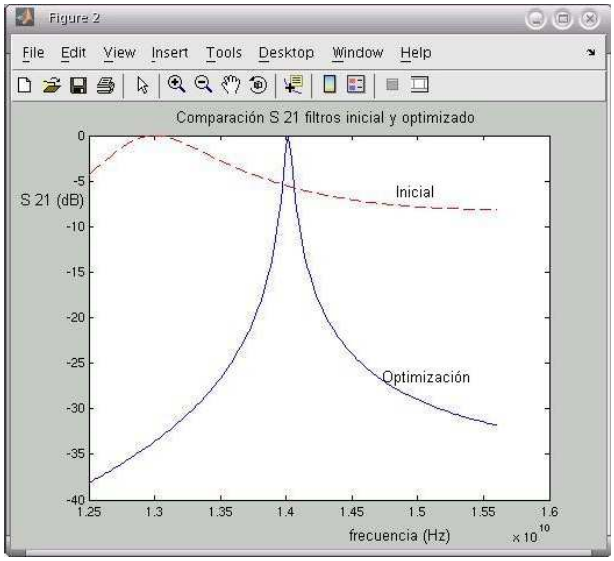

Figura 48. Respuesta del filtro optimizado por gradiente

A primera vista parece que la optimización por gradiente no consigue una mejora sustancial respecto del último resultado, pero si analizamos la región de paso detalladamente, se observa cómo este filtro se ajusta más a nuestras necesidades:

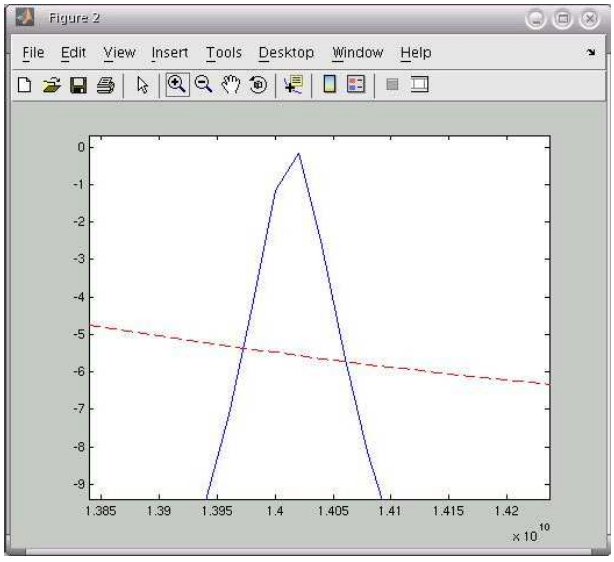

Figura 49. Detalle de la respuesta del filtro optimizado por gradiente

La gráfica no presenta una buena definición, debido a que se crea a partir de los 100 puntos para todo el intervalo de frecuencias que se especificaron al comienzo, por lo que dentro de la banda de paso sólo hay unos pocos. Aún así, la bondad del resultado es evidente. Para realizar la optimización por gradiente se emplea la función *fminunc* de *Matlab*, la cual busca el mínimo de una función siguiendo una variación del método de Quasi-Newton. Esta variación se basa en la fórmula BFGS para aproximar el Hessiano y en combinar la interpolación cuadrática y cúbica entre puntos. Se ha limitado el número de evaluaciones del proceso a 100, para que no tarde demasiado, como se explicó en el capítulo dedicado a los procesos de minimización.

La segunda opción que se nos ofrece consiste en seleccionar un individuo y generar a partir de él un nuevo proceso evolutivo. Apretando sobre el botón **Seleccionar** se accede a una gráfica que muestra las puntuaciones de todos los individuos, ordenadas de mayor a menor, de las distintas generaciones:

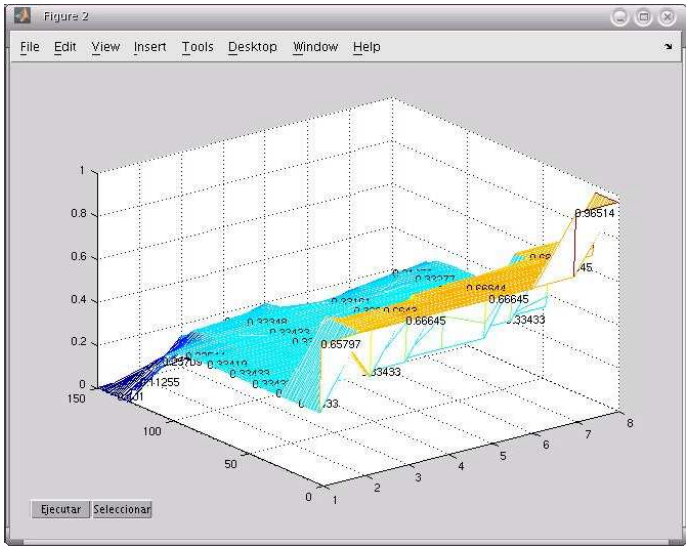

Figura 50. Ventana de Selección de individuos

De esta manera podemos elegir como prototipo cualquier individuo generado con anterioridad, aunque no sea el de mayor puntuación. Pulsando con el ratón sobre un punto de la superficie de puntuaciones, preseleccionamos dicho elemento, y se muestra una etiqueta identificativa del mismo: generación a la que pertenece, orden que ocupa dentro de su generación y puntuación. Para conocer más del elemento en cuestión, con el botón **Ejecutar** podemos conocer su respuesta en frecuencia y dimensiones físicas. Si las características del individuo nos convencen, será el punto de partida para la nueva evolución. Se pueden cambiar las variables de optimización o parámetros del algoritmo (número de iteraciones, método de selección, etc) mediante el botón **Variables**. En este ejemplo se mantuvieron todas las variables y opciones como estaban. Se seleccionó el segundo elemento de la última iteración, que presenta una puntuación de 0′9651. Sólo queda pulsar el botón **Seleccionar**; el algoritmo genético comienza automáticamente. Tras 8 generaciones, el resultado obtenido presenta una puntuación de 0′9896, lo cual es muy satisfactorio. La respuesta de este filtro es la siguiente:

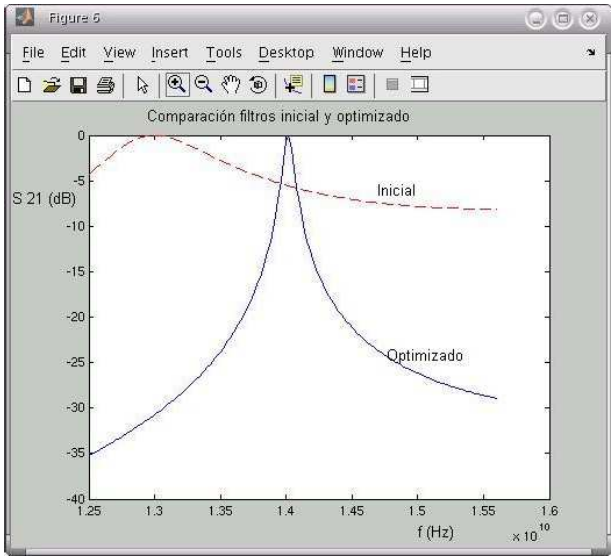

Figura 51. Respuesta del filtro optimizado

Puede observarse que es muy similar a la conseguida mediante el método de optimización por gradiente. Procediendo como antes, ampliamos la parte central de la imagen, para comparar ambas respuestas en la banda de paso:

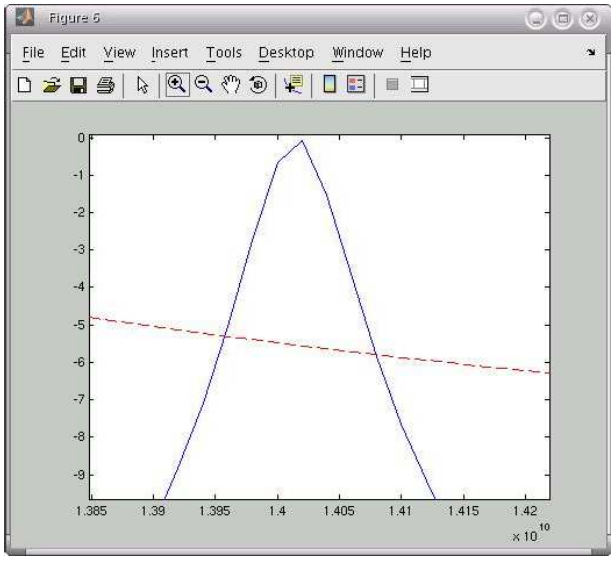

Figura 52. Detalle de la respuesta del filtro optimizado

En este caso vemos cómo el ancho de banda de la señal a -3 dB es un poco mayor del buscado (desde 13′955 hasta 14′040 GHz). El hecho de su alta puntuación se debe a que todas las muestras del ancho de banda están por encima de los -3 dB especificados; similarmente, prácticamente todas las muestras de la banda de rechazo caen por debajo de -16 dB. La diferencia entre ambas respuestas está en la banda de tránsito: una presenta más pendiente que la otra. Así, entre individuos con puntuaciones muy altas (p.e. superiores al 97 %), la elección de uno u otro puede hacerse manualmente atendiendo a parámetros que este programa no evalúa, como la pendiente en la banda de tránsito.

La evolución seguida por el algoritmo genético en este último proceso ha sido fructífera, como muestra la gráfica:

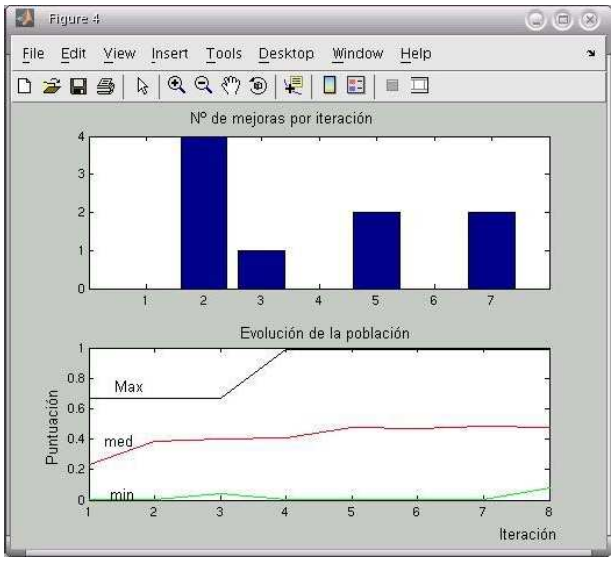

Figura 53. Evolución de la nueva población

También vemos cómo la primera generación creada a partir del filtro seleccionado presentaba una distribución uniforme de sus genes:

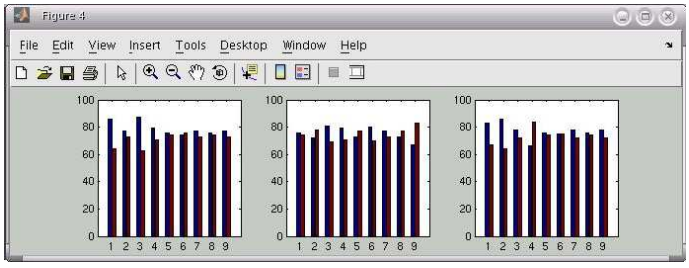

Figura 54. Distribución uniforme de la nueva población

Tras 8 iteraciones, el desgaste genético es claro en las 3 variables:

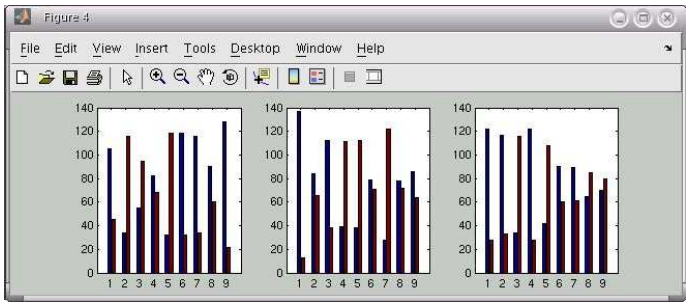

Figura 55. Distribución de la población en la 8ª iteración

A la vista de los buenos resultados obtenidos con ambos métodos, damos por concluido el presente ejemplo de optimización. A continuación se muestran los esquemas de los filtros encontrados en las distintas partes del ejercicio, para que se comparen y se vea la evolución seguida:

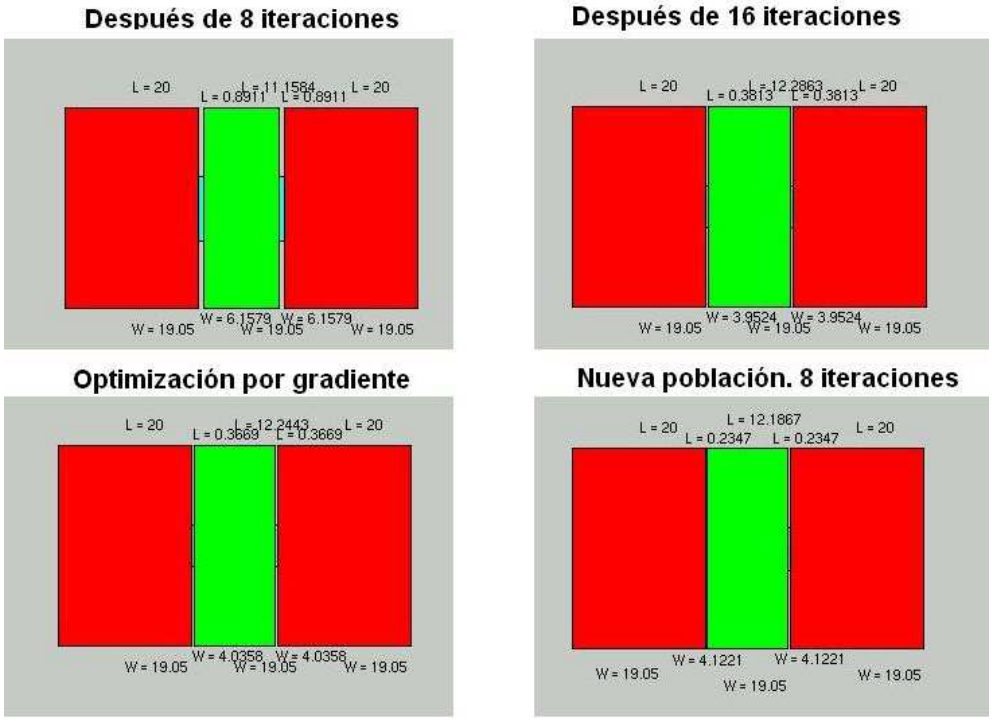

Figura 56. Comparación de las dimensiones de los filtros obtenidos

Comparando el primer filtro con la figura inicial observamos que la longitud de la cavidad resonante aumenta ligeramente; pero el mayor cambio lo sufren las discontinuidades inductivas, reduciendo notablemente tanto su longitud como anchura. El proceso continúa con las siguientes iteraciones; la longitud de la cavidad vuelve a aumentar poco más de un milímetro, mientras que la anchura de los escalones se reduce en $\frac{1}{3}$ y su longitud lo hace algo más de la mitad. Los cambios originados en la última parte del diseño son más sutiles. Las dimensiones para el filtro encontrado mediante la optimización por gradiente son similares a las obtenidas tras cambiar de prototipo. Podríamos decir que en las primeras iteraciones se realiza un primer ajuste y que mediante los métodos de gradiente y generación de población a partir de un individuo, se obtiene un ajuste fino de las dimensiones.

4.2. Filtro de Orden 2 a 10 GHz

Vamos a ver ahora cómo reacciona el programa ante un problema un poco más difícil: la optimización de un filtro de orden 2. Las características de este filtro son:

$$\begin{aligned} \text{Banda de Paso: } 3\ dB\ en\ 10{,}50 - 10{,}55\ GHz \\ \text{Banda de Rechazo: } 16\ dB\ fuera\ de\ 10 - 10{,}60\ GHz \end{aligned} \tag{73}$$

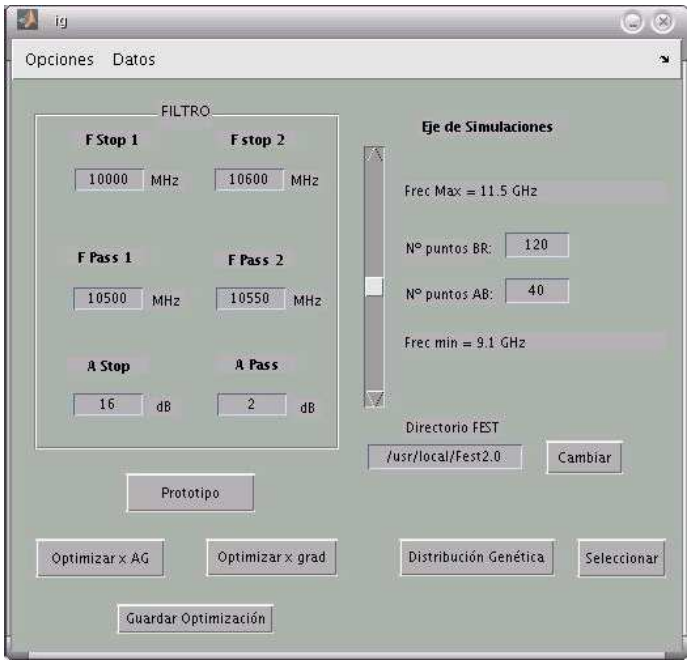

Figura 57. Ventana principal del programa. Nuevo filtro

En este caso la respuesta del modelo teórico del que partimos es:

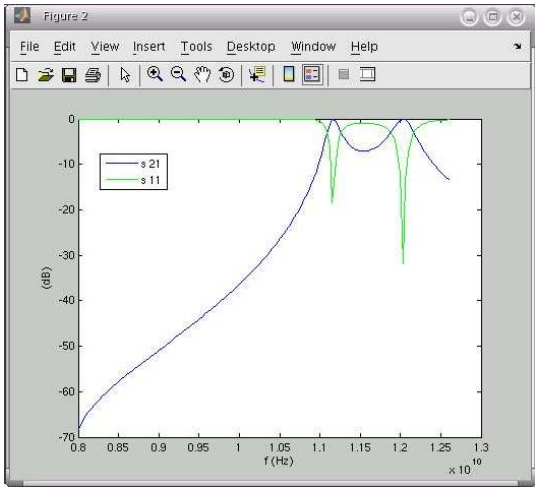

Figura 58. Respuesta del prototipo inicial

Podemos observar el rizado propio de un filtro de orden dos en la banda de paso, centrada en este caso en 11.5 GHz y con un ancho de banda de 1 GHz. El intervalo de frecuencias se ha ampliado para incluir esta banda entera.

La configuración escogida para la optimización fue:

- Población: 100 individuos
- Número de iteraciones: 12
- Probabilidad de mutación: 6 %
- Número de bits por variable: 8
- Base del algoritmo: 2
- Método de selección: Torneo

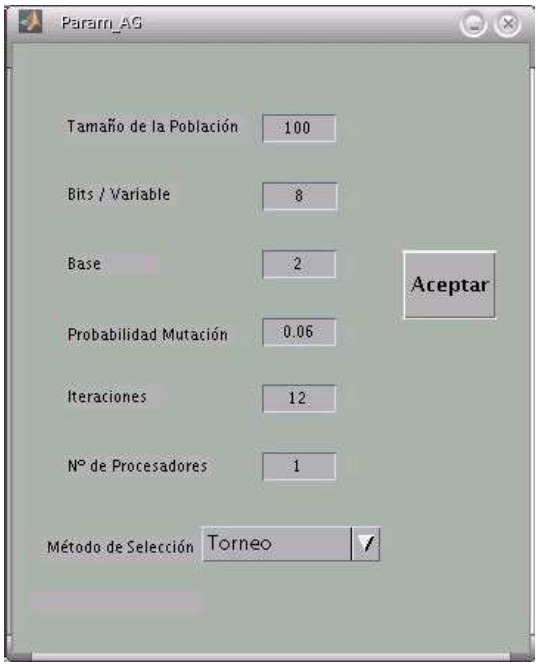

Figura 59. Parámetros de configuración del Algoritmo Genético

Si abrimos la ventana de *Variables de Simulación* vemos que podemos seleccionar más elementos que antes. Esto se debe a que ahora se trata de un filtro de orden 2, por lo que tenemos 2 cavidades resonantes y 3 discontinuidades para elegir. Seleccionamos la anchura y longitud de las cavidades y la anchura de los escalones, tal como hicimos antes:

Figura 60. Variables de optimización

Con estos parámetros ejecutamos la optimización, obteniendo como resultado:

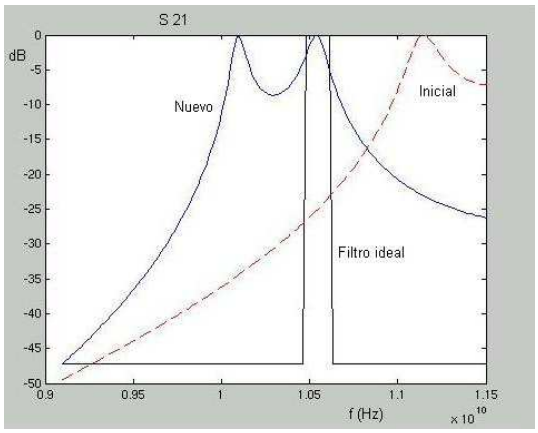

Figura 61. Respuesta del filtro obtenido

En la gráfica se puede apreciar cómo se desplaza la banda de paso hasta su posición ideal, 10.5 GHz, pero sigue teniendo un ancho de banda mayor al deseado. El algoritmo genético no ha sido capaz en este ejemplo de optimizar mucho el diseño original.

Si analizamos las gráficas siguientes comprobamos que tan solo se ha producido un modelo mejor que el primero, que ha tardado 7 iteraciones en aparecer. Los cruces que se efectuaron en los 6 primeros ciclos sólo dieron como resultado diseños iguales o peores que el de partida. Con estos datos concluimos que el problema está en la configuración genética y que variándola será más probable hallar soluciones que se aproximen mejor a los requisitos iniciales.

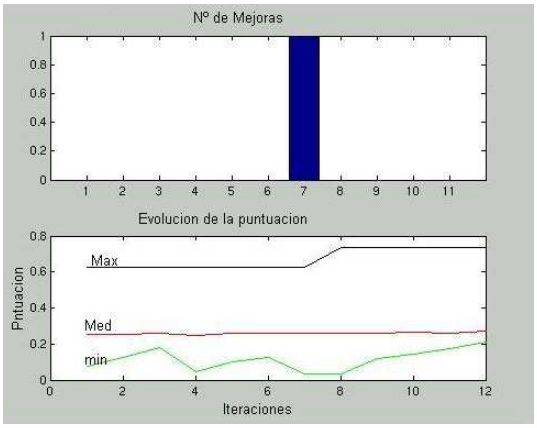

Figura 62. Evolución de la población en 12 iteraciones

La población inicial tenía una distribución uniforme:

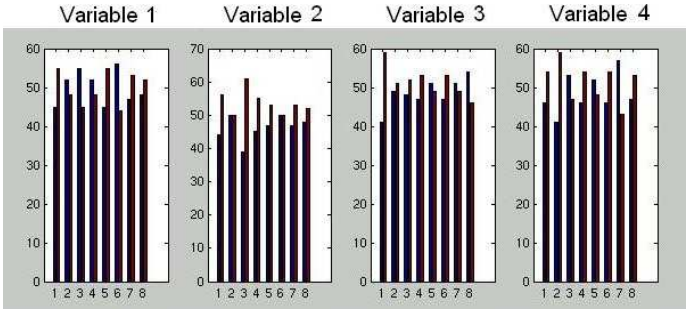

Figura 63. Distribución genética uniforme en la población inicial

En la sexta iteración la población ha cambiado sustancialmente, predominando un valor sobre el otro en algunos bits de las variables:

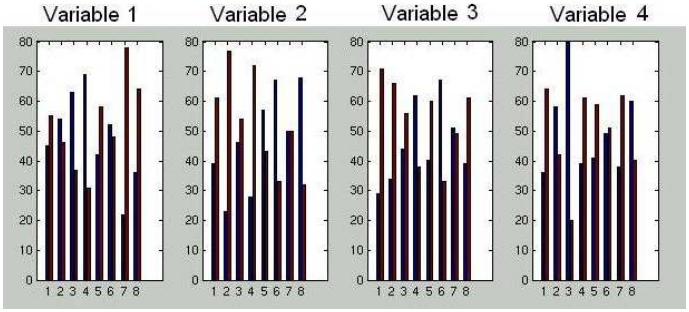

Figura 64. Distribución genética en la 6ª iteración

En la última población está más acentuada la tendencia:

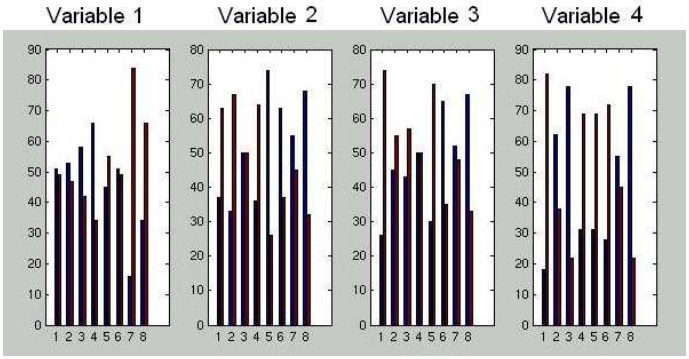

Figura 65. Distribución genética en la 12ª iteración

Hay que resaltar que algunas variables convergen antes que otras. Cuando el algoritmo elegido permite la repetición de un mismo individuo, puede suceder que estos clones ocupen un porcentaje importante de la población (por ejemplo, el 20 %), limitando su capacidad de evolución. Si esto sucede, la gráfica anterior presentaría valores más definidos.

4.3. Filtro de Orden 3 a 12 GHz

El siguiente ejemplo ilustra cómo la parte aleatoria del algoritmo genético empleado puede ser determinante para el resultado final. Para ello, intentaremos resolver un mismo problema dos veces, con la misma configuración. El filtro escogido está centrado en 12 GHz:

$$\begin{aligned} \textit{Banda de Paso}: \ -3\ dB\ en\ 12{,}00 - 12{,}05\ GHz \\ \textit{Banda de Rechazo}: \ -20\ dB\ fuera\ de\ 11{,}95 - 12{,}10\ GHz \end{aligned} \tag{74}$$

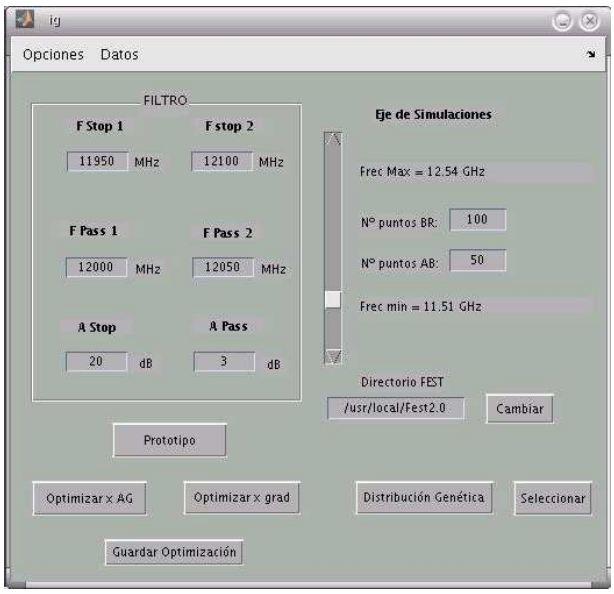

Figura 66. Ventana principal del programa

El modelo teórico para un filtro con estas especificaciones es de orden 3:

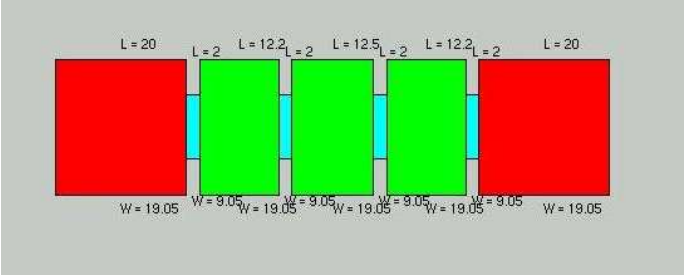

Figura 67. Dimensiones del filtro prototipo. Orden 3

La función de transferencia de este filtro está representada en la imagen siguiente:

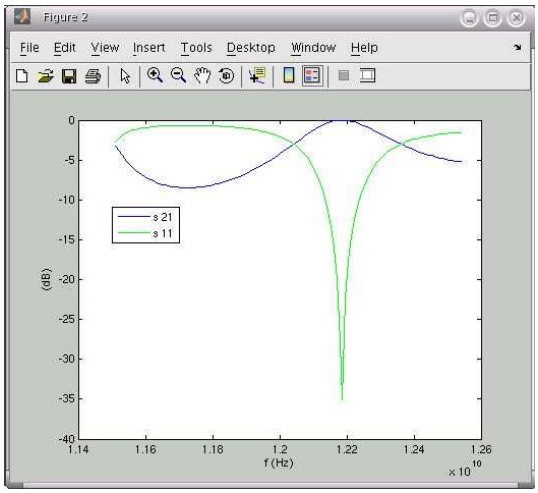

Figura 68. Respuesta en frecuencia del modelo teórico

Vemos que el eje de frecuencias escogido parece demasiado pequeño para este filtro, pero es suficientemente grande para albergar la banda de paso y converger a la banda de rechazo teórica.

Trabajaremos sobre una población de 100 individuos con 12 iteraciones. Como en casos anteriores, manejaremos cuatro variables: longitud y anchura de las cavidades resonantes y longitud y anchura de las intersecciones, mediante la opción *All* que ya vimos. Los resultados obtenidos en ambas simulaciones se muestran a continuación:

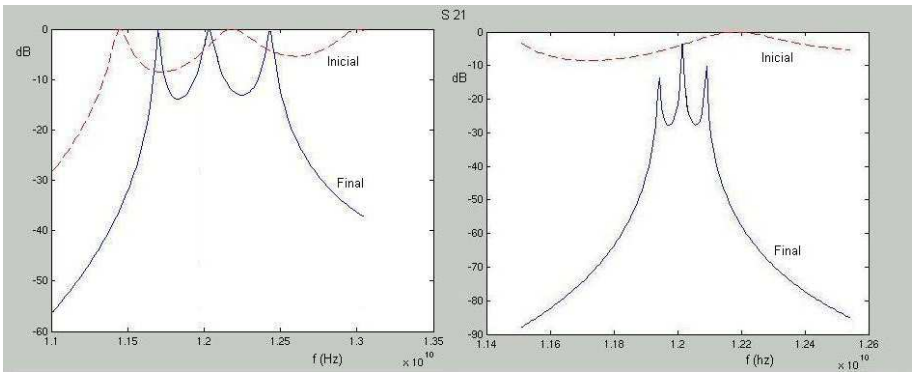

Figura 69. Comparación de los filtros obtenidos en 2 simulaciones

Como puede observarse, estas respuestas son claramente distintas entre sí. La primera optimización presenta menos atenuación en la banda de paso, pero es menos selectiva que la segunda. Hay que resaltar que en ambas se ha conseguido centrar la banda de paso en 12 GHz, como se quería.

Partimos de poblaciones similares en ambos casos, pero el desarrollo ha sido diferente. En el primer caso, se encontró un diseño en la segunda iteración cuyas características no se superaron después. Este individuo supuso una gran mejora del modelo inicial:

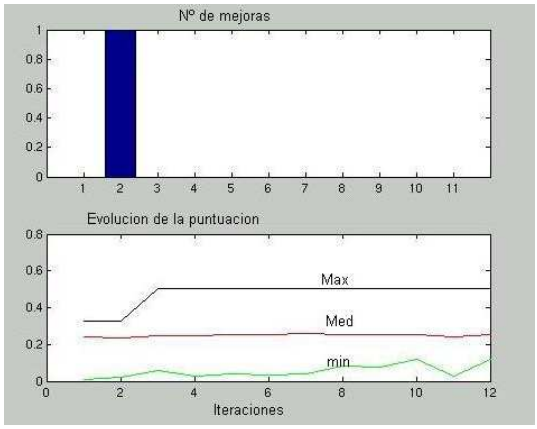

Figura 70. Evolución de la población para la primera simulación

Se podría pensar que la población se ha saturado de clones de ese individuo, pero entonces habría aumentado significativamente la puntuación media de la población, hecho que no sucede. Podemos asegurar, por tanto, que la población mantiene su riqueza genética, aunque no haya encontrado diseños más optimizados. En la segunda simulación, el programa obtuvo sucesivas mejoras, aunque de menor índole:

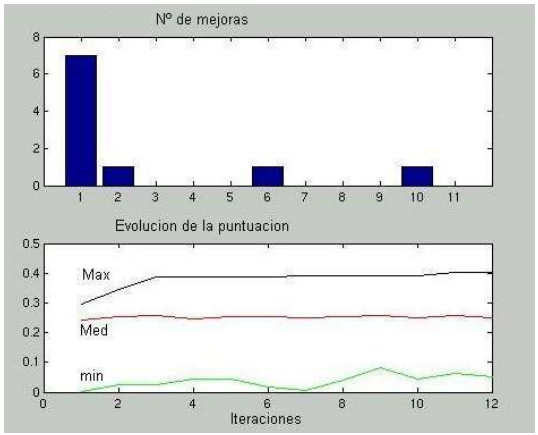

Figura 71. Evolución de la población para la segunda simulación

Queda demostrado que las condiciones iniciales del problema no determinan el resultado de su optimización. Si el resultado obtenido en una optimización no ha sido satisfactorio, una opción que no se debe descartar es la de repetir el proceso, aunque no se realicen cambios de configuración.

4.4. Filtro de Orden 2 a 9 GHz

Para comparar los distintos métodos de selección de la población que puede seguir un algoritmo genético, a continuación optimizaremos un filtro mediante estos tres métodos: **ruleta, torneo y torneo probabilístico**. Las especificaciones para este filtro son:

$$\begin{aligned} \text{Banda de Paso: } & -3\ dB\ en\ 9{,}05 - 9{,}09\ GHz \\ \text{Banda de Rechazo: } & -16\ dB\ fuera\ de\ 9{,}00 - 9{,}20\ GHz \end{aligned} \tag{75}$$

Introducimos estos datos en la ventana principal del programa:

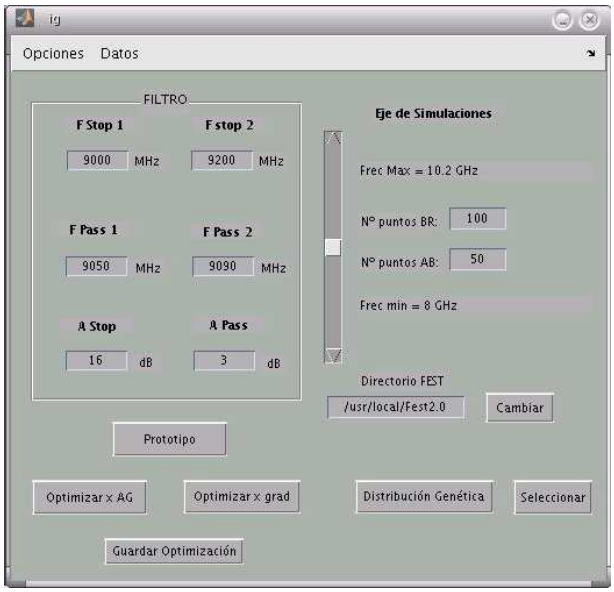

Figura 72. Ventana principal del programa con los datos del nuevo filtro

Se obtiene un filtro de orden 2:

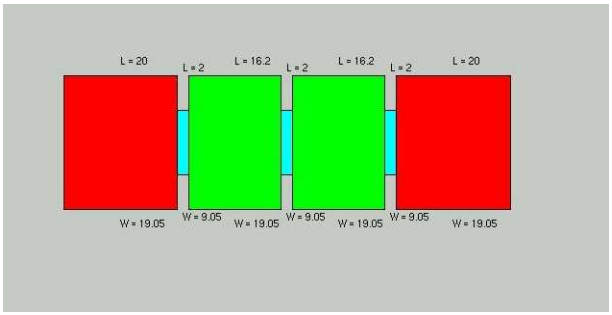

Figura 73. Dimensiones del filtro teórico inicial

Realizaremos 8 iteraciones sobre una población de 100 individuos, con una probabilidad de mutación de 6 %. Primero se utilizará la *ruleta*. Cuando seleccionamos esta opción, tenemos que especificar el número de ruletas con que queremos trabajar. Dada la población de 100 muestras, elegimos 8 ruletas: 7 serán de 12 individuos y la octava de 16. Una ruleta no conviene que maneje pocos individuos, porque aquellos que tengan mayores puntuaciones saldrán elegidos demasiadas veces y la siguiente generación perderá mucha riqueza genética, aumentando el riesgo de saturación.

Por el contrario, si se incluyen muchas muestras en una misma ruleta, se distorsiona la diferencia de puntuación entre individuos, perdiendo su ventaja aquellos mejor dotados.

Tras las simulaciones, se obtiene como resultado el modelo que aparece en la figura:

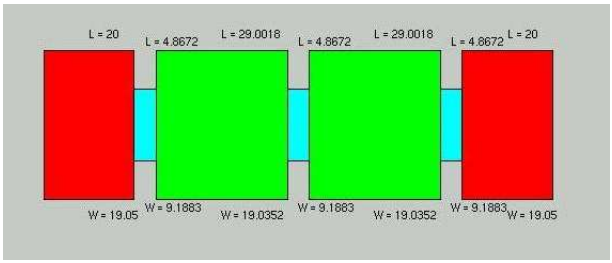

Figura 74. Dimensiones del filtro obtenido tras las simulaciones

La respuesta de este filtro mejora considerablemente la del modelo teórico:

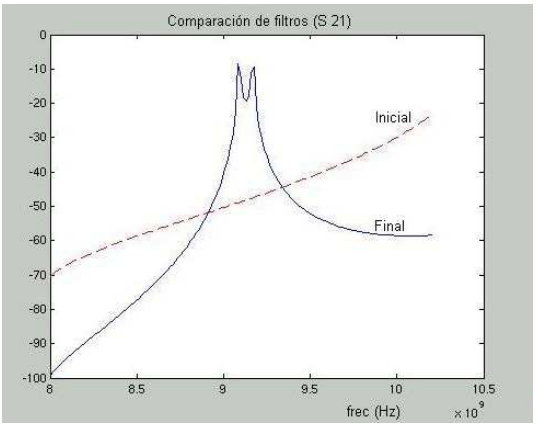

Figura 75. REspuesta del filtro obtenido tras las simulaciones

Sin embargo, realizando un zoom sobre la imagen, vemos que todavía no cumple con las especificaciones que buscamos:

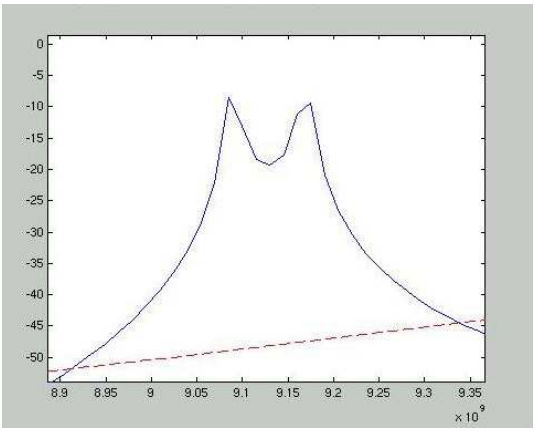

Figura 76. Zoom de la respuesta del filtro obtenido

La evolución seguida por el algoritmo en este caso, hace suponer que es apropiado cambiar algún parámetro:

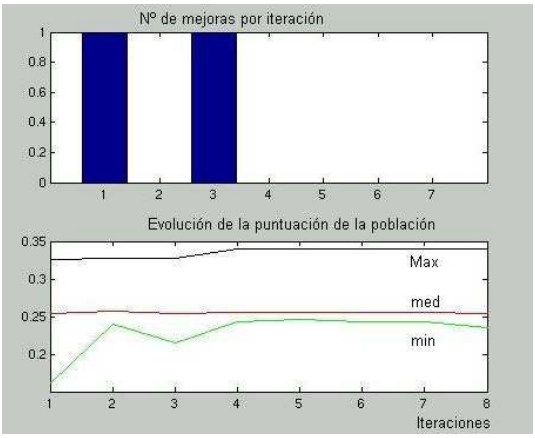

Figura 77. Evolución de la población en 8 iteraciones

Por ello, en vez de empezar desde el principio un nuevo proceso de optimización con otro método, vamos a continuar a partir de ésta. Cambiamos la configuración, eligiendo esta vez el método del *torneo*. Como estamos continuando sobre la simulación que acabamos de realizar, la nueva población inicial será la última del proceso anterior:

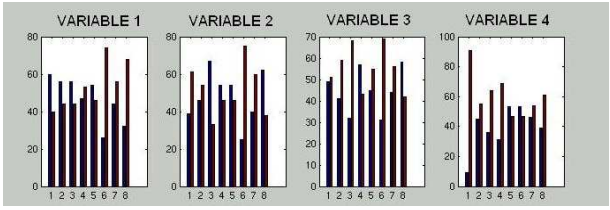

Figura 78. Distribución genética inicial. Ultima población anterior.

Esto podemos hacerlo porque no se ha cambiado ni el tamaño de la población ni las variables a analizar. Esta posibilidad de poder continuar una simulación cambiando parámetros permite tener mayor control sobre el proceso evolutivo. Es aconsejable realizar varias simulaciones concatenadas, de pocas iteraciones cada una, en vez de realizar una sola larga. De esta forma, podemos detectar en tiempo real cualquier problema, como una convergencia prematura o un estancamiento de la población, y tomar medidas para evitarlo.

Realizamos la nueva simulación y obtenemos un filtro, cuya respuesta es mejor:

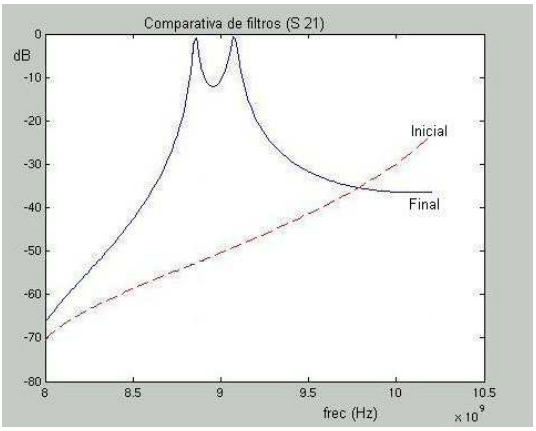

Figura 79. Respuesta del filtro de la nueva simulación

El esquema de este filtro no se parece mucho al anterior:

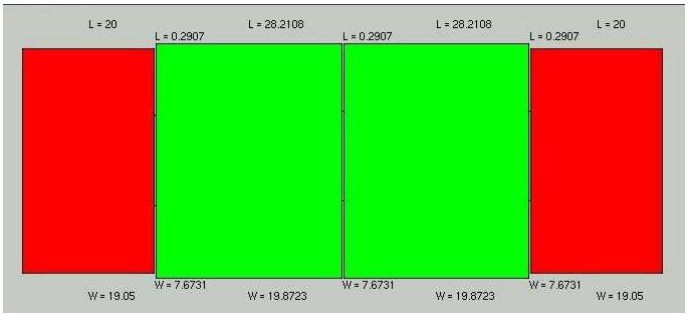

Figura 80. Dimensiones del nuevo filtro

Con la configuración actual, la población ha evolucionado mejor para este problema que en su primera simulación, aumentando la función de evaluación considerablemente hacia el final del proceso:

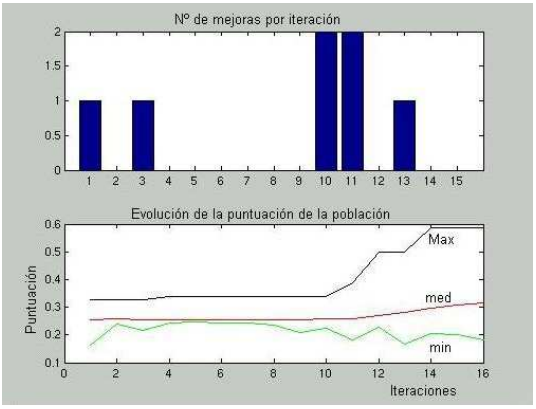

Figura 81. Evolución de la población en las 16 iteraciones totales

La población en la última iteración aparece bastante focalizada en un área, como muestra la distribución binaria de las variables:

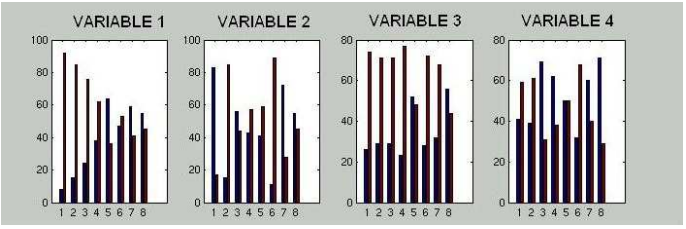

Figura 82. Distribución genética de la última iteración

Por esta razón, la siguiente simulación no se realizará a partir de esta población, sino que se creará una nueva, uniforme.

Queda probar el método del *torneo probabilístico*. Seleccionemos esta opción en la ventana de Parámetros. El valor de *Factor* hace referencia a la variable **T**, descrita en la ecuación 35. Valores normales para esta variable están entre 1 y 2, aunque admite cualquier valor positivo. En este caso se ha elegido 1,5. Con esta configuración ejecutamos las simulaciones. El filtro que se consigue es, de nuevo, bastante diferente del original:

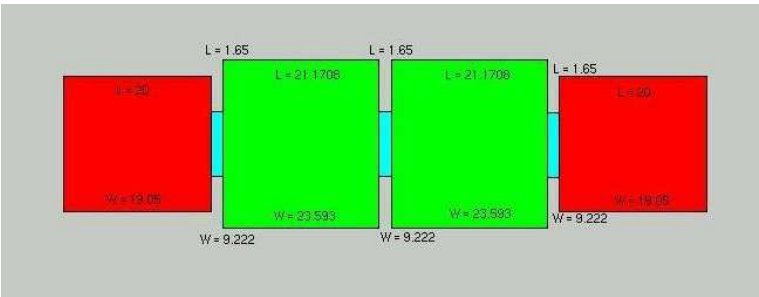

Figura 83. Dimensiones del filtro obtenido por torneo probabilístico

Su respuesta en frecuencia se aproxima al filtro deseado:

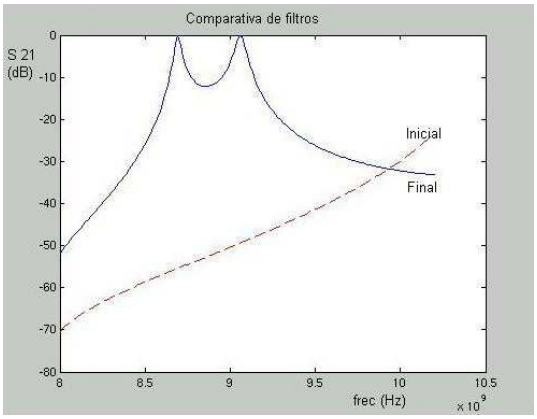

Figura 84. Respuesta del nuevo filtro obtenido

De hecho, si aumentamos la imagen, vemos que uno de los dos extremos de la banda de paso de este filtro coincide con la banda de paso objetivo. Dado que el rizado que presenta supera los 3 dB, se produce una especie de banda de paso justo donde queríamos, aunque tenemos otra en 8,7 GHz:

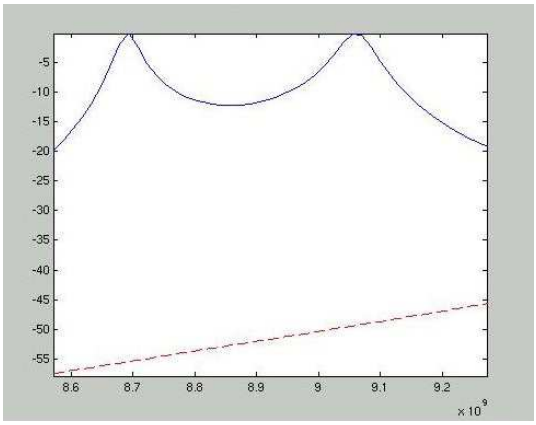

Figura 85. Detalle de la respuesta en frecuencia de filtro

Esta es la razón de que este filtro obtenga una puntuación más alta que los anteriores:

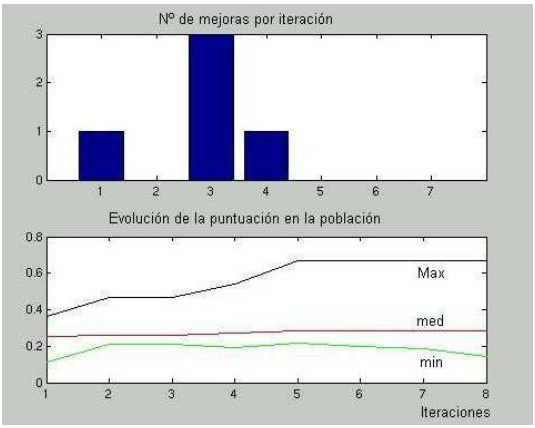

Figura 86. Evolución de la población en 8 iteraciones

En la distribución de bits de la población en la última iteración aparece poco definida todavía, por lo que se podría seguir iterando sobre ella; pero el hecho de que en las últimas 4 iteraciones no se haya encontrado ningún diseño más apto hace suponer que sería necesario algún cambio de configuración.

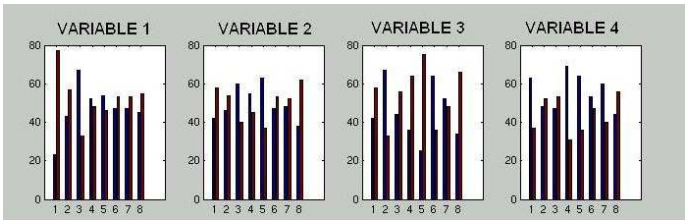

Figura 87. Distribución genética en la 8ª iteración

4.5. Filtro de Orden 2 a 16 GHz

Vamos a ver un ejemplo que ilustra la dificultad en el diseño de filtros de orden 2. Si con los filtros de orden 1 primero centrábamos la banda de paso y después

ajustamos el ancho de banda, la estrategia a seguir con los de 2º orden es similar, pero aparece el problema del rizado en la banda de paso. El rizado aparece en todo filtro de orden superior a 1, y es consecuencia del acoplo entre las sucesivas cavidades resonantes. Si este rizado es muy pronunciado, el algoritmo puede converger a un filtro cuyo pico inferior o superior coincida con la banda de paso buscada. Esto es lo que sucede en el siguiente caso:

Supongamos que queremos realizar un filtro centrado en 16′250 GHz.

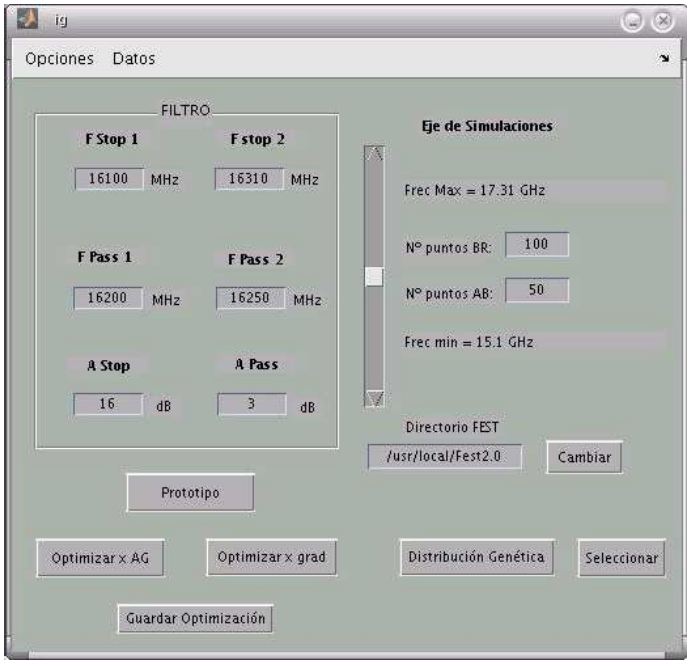

Figura 88. Introducción de datos del filtro en la ventana principal

Para cumplir estas especificaciones de diseño se requiere una estructura de 2º orden. El filtro que sirve de base para iniciar el proceso evolutivo es el de la figura:

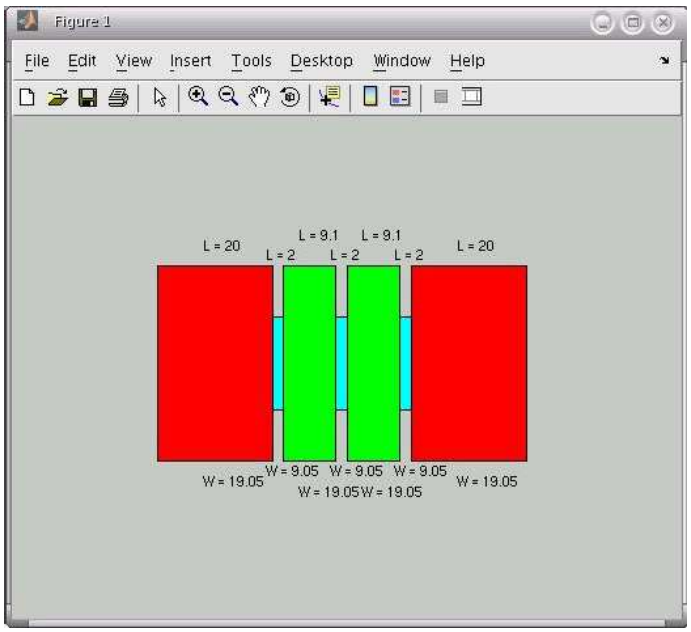

Figura 89. Dimensiones del filtro modelo teórico

La respuesta de este filtro está muy apartada de la deseada. Como variables para el proceso se escogieron, como en otros casos, la longitud de las cavidades y la anchura de las discontinuidades:

Figura 90. Variables a optimizar

Se iteró 8 veces sobre una población de 120 individuos. El programa da como resultado un filtro cuya respuesta aparece contrapuesta a la del filtro primero (en línea discontinua):

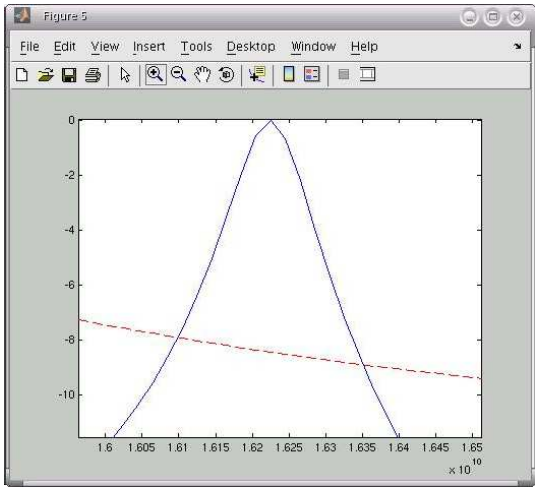

Figura 91. Respuesta del filtro resultado de 8 iteraciones

Se podría pensar que este resultado es satisfactorio como primera aproximación, y que con más iteraciones se podría reducir su ancho de banda a 3 dB de 100 MHz a la mitad, al igual que en otros ejemplos ya vistos. Sin embargo, aumentando el eje de frecuencias, el comportamiento del filtro ya no parece tan bueno:

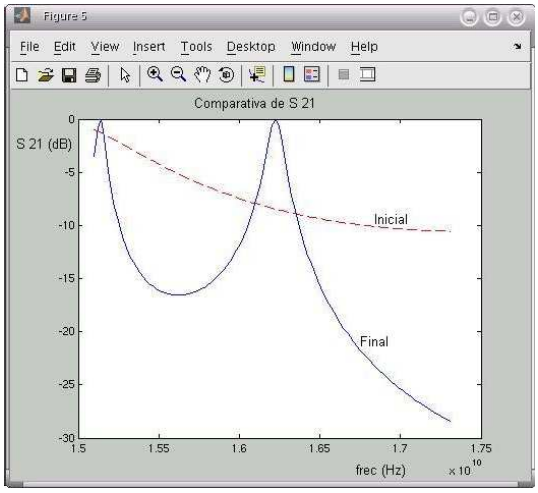

Figura 92. Respuesta completa del filtro resultado

Lo que realmente tenemos es un filtro con un rizado de 16 dB y un *ancho de banda* de 1 GHz, que se aleja bastante de lo que queríamos conseguir. Si observamos la estructura que conforma dicho filtro comprobamos que la abertura central es mayor que las laterales, lo que provoca este comportamiento:

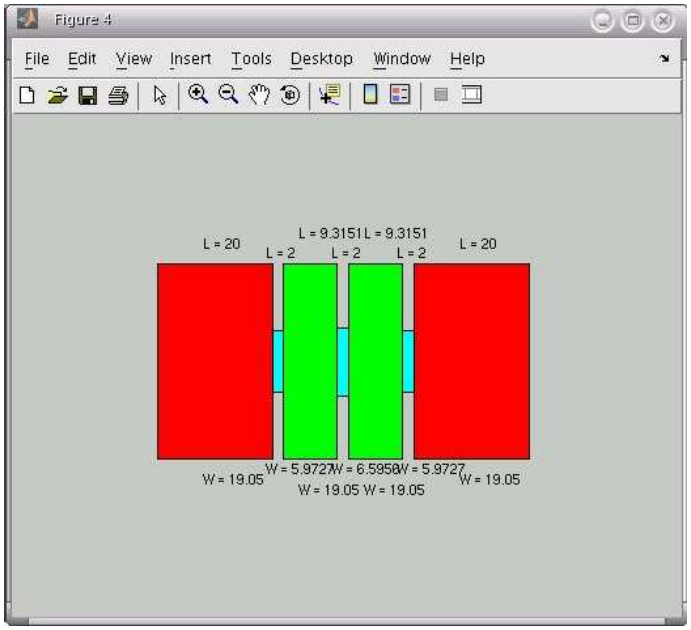

Figura 93. Dimensiones del filtro anterior

En lugar de corregir la longitud de las cavidades resonantes, el programa se ha centrado en la abertura de acoplo. La explicación de que este filtro obtenga una buena puntuación (de 0'75) se debe a que su respuesta en la banda de paso (16'200 a 16'250 GHz) es correcta, coincidiendo con el pico superior del rizado; en la banda de rechazo el filtro presenta igualmente una buena respuesta, excepto el pico inferior del rizado (en la región de 15'200 GHz). Supongamos, para simplificar, que la puntuación correspondiente a la banda de paso se acerca a 1 y la de la banda de rechazo es 0'5: la media es 0'75. Si se hubiese escogido un intervalo de frecuencias mayor, el peso del pico inferior sobre la evaluación habría sido mayor, ya que aparecería entero y no cortado, disminuyendo la puntuación final. Si, por el contrario, se hubiera seleccionado un intervalo de estudio menor, este pico no quedaría reflejado en la evaluación, resultando en una mayor puntuación para el filtro. Podríamos, por tanto, pensar que cuanto mayor sea la ventana de estudio más fiable es el resultado obtenido; pero a mayor intervalo necesitaremos más puntos de resolución para mantener la misma definición en la respuesta de los filtros, lo cual incrementa el tiempo necesario para llevar a cabo las simulaciones. De ahí la importancia de encontrar un equilibrio entre el tamaño de la ventana de estudio y el tiempo necesario para realizar dichas simulaciones.

4.6. Filtro de Orden 2 a 9 GHz

El siguiente ejemplo ilustra el diseño de un filtro de orden 2. Las especificaciones para este filtro son:

$$\begin{aligned} \text{Banda de Paso: } 3\ dB\ en\ 9{,}05 - 9{,}09\ GHz \\ \text{Banda de Rechazo: } 16\ dB\ fuera\ de\ 9 - 9{,}2\ GHz \end{aligned} \tag{76}$$

El modelo inicial es el siguiente:

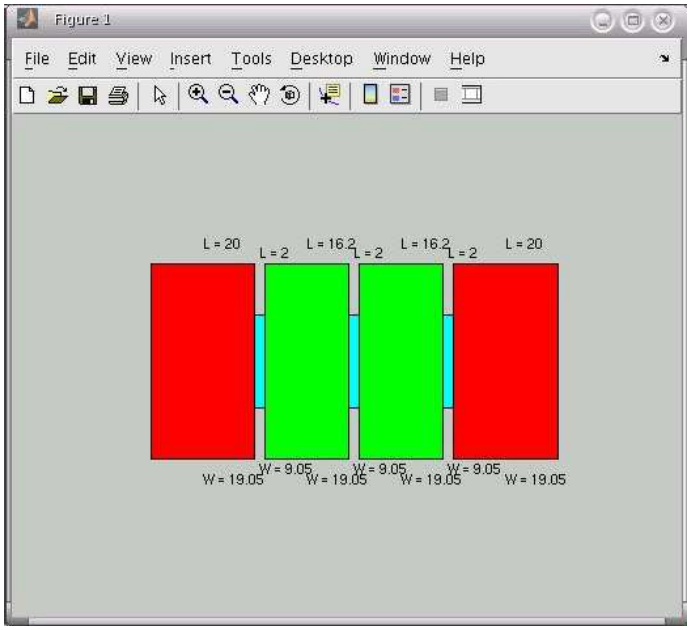

Figura 94. Dimensiones del filtro modelo inicial

La respuesta de este modelo queda muy alejada de lo esperado. Vamos a intentar optimizar 3 variables al mismo tiempo: la longitud de las cavidades resonantes y la anchura de los escalones. Al tratarse de un filtro de orden 2 simétrico, las longitudes de las 2 cavidades han de ser igual entre sí; lo mismo sucede con los escalones de los extremos (1 y 3); la tercera variable hace referencia al segundo escalón, el que controla el acoplo entre las cavidades. Los parámetros que rigen el algoritmo genético se muestran a continuación:

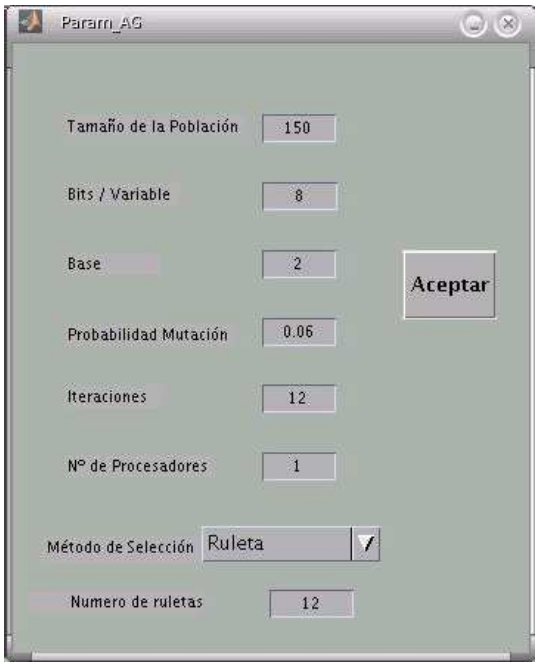

Figura 95. Parámetros de configuración del A.G.

Ejecutamos las 8 iteraciones. Para analizar el estado en el que se encuentra la población, en lugar de ver las gráficas de distribución genética de sus variables, vamos a visualizar las puntuaciones de los filtros:

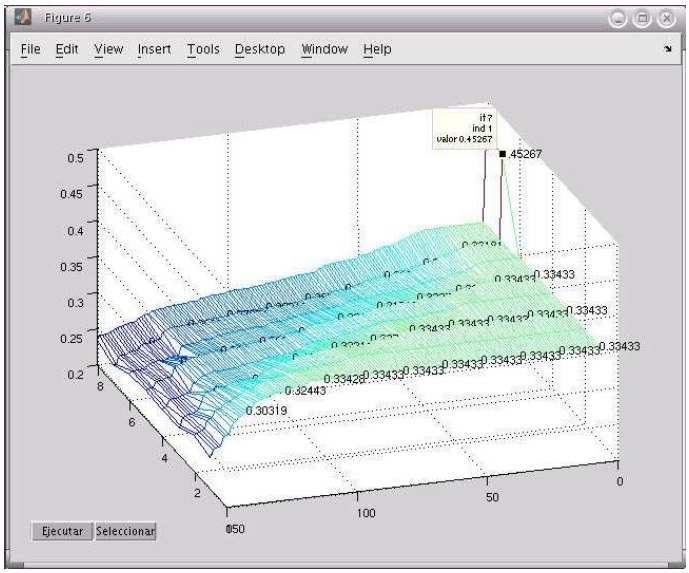

Figura 96. Superficie de puntuaciones de toda la población

Se observa que las puntuaciones no son buenas,tienen valores bastante bajos; además, la superficie es muy plana, lo cual es muestra de que hay muchos individuos iguales o similares. Por tanto, es necesario cambiar de población antes de continuar. Seleccionamos un individuo y repetimos el proceso. En este caso los resultados son mejores, lográndose individuos con puntuación de 0′8, como puede verse en la gráfica:

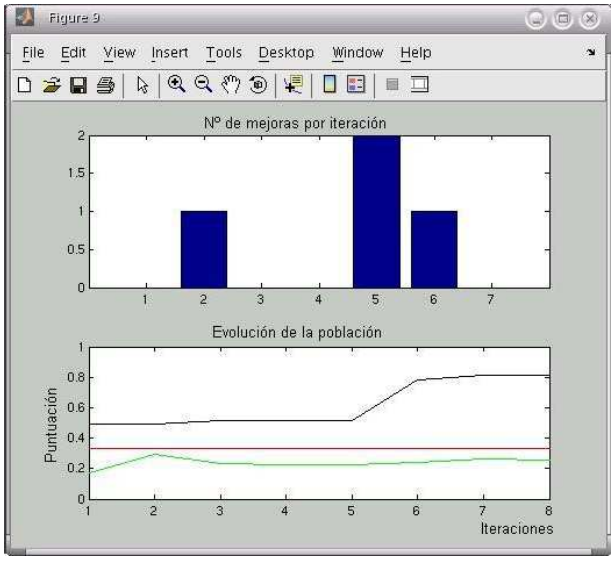

Figura 97. Evolución de la puntuación de la población

El filtro así obtenido tiene la siguiente forma:

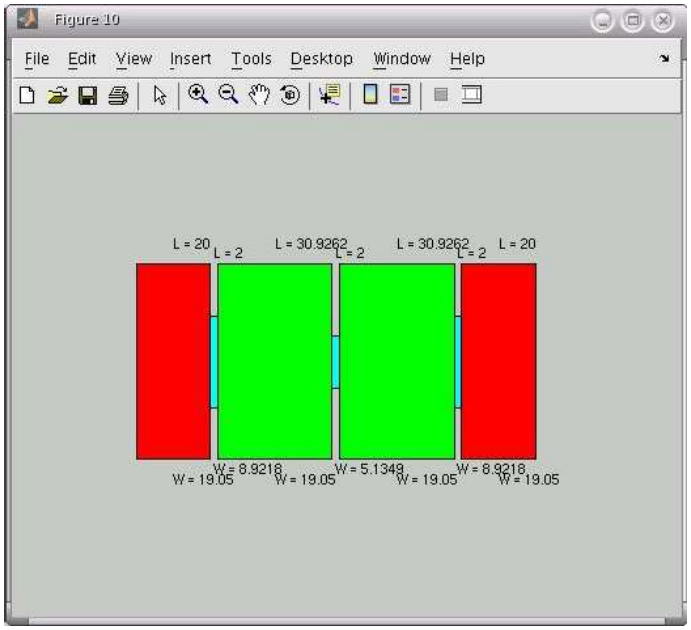

Figura 98. Dimensiones obtenidas tras 8 iteraciones

Nótese el gran cambio en la longitud de las cavidades, que explica porqué la respuesta del modelo primero está tan alejada del objetivo. Por último, como ajuste fino, se realiza una optimización por gradiente. Las dimensiones del filtro optimizado aparecen en el siguiente esquema:

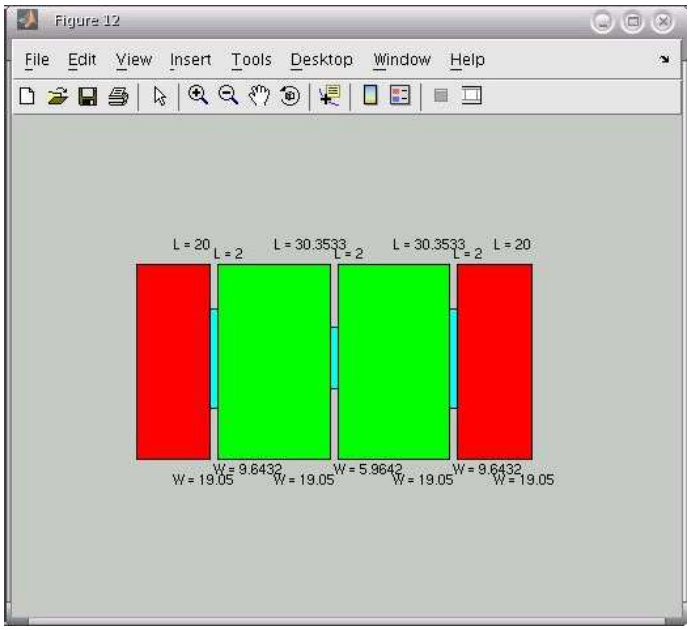

Figura 99. Dimensiones del filtro optimizado por gradiente

Los escalones son un poco más anchos que antes y las cavidades se mantienen prácticamente igual.

A continuación se muestra una comparación de las respuestas en frecuencia de los distintos filtros obtenidos a lo largo del proceso. La línea discontinua (1)

corresponde al modelo inicial, la azul (2) al paso intermedio, la marrón (3) al filtro final y la negra (4) a la optimización por gradiente:

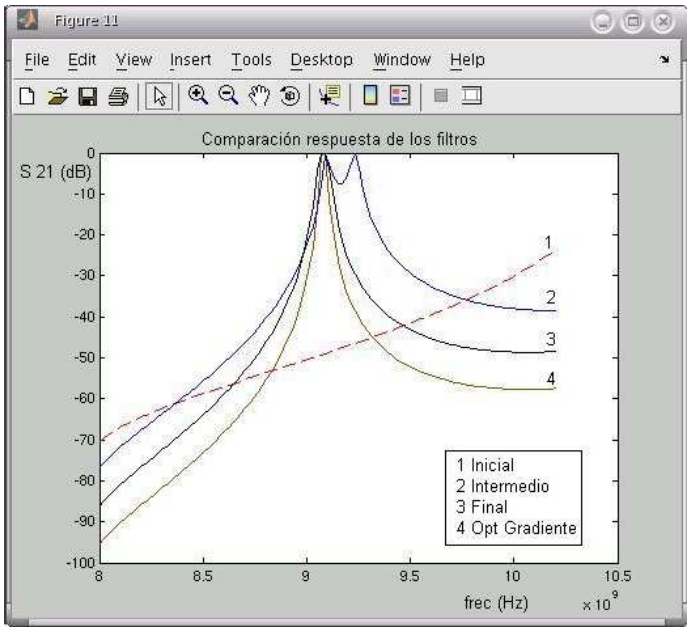

Figura 100. Comparación de los filtros obtenidos durante el proceso

Si nos centramos en los dos últimos, vemos cómo tanto el filtro optimizado genéticamente como su posterior mejora por gradiente cumplen las especificaciones referentes a la banda de rechazo; en cuanto a la banda de paso, son muy selectivos, quizás demasiado para el segundo caso:

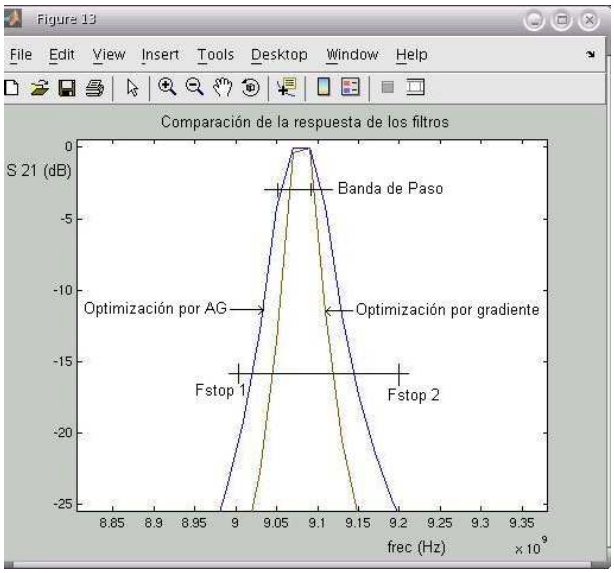

Figura 101. Detalle de los filtros optimizados por A.G. y por Gradiente

4.7. Filtro de Orden 2 a 11 GHz

Este ejemplo trata del diseño de un filtro en la banda de 11 GHz. Sus caracaterísticas concretas son:

$$\begin{array}{c} \text{Banda de Paso: } 3\ dB\ en\ 11{,}05 - 11{,}09\ GHz \\ \text{Banda de Rechazo: } 18\ dB\ fuera\ de\ 11 - 11{,}2\ GHz \end{array} \tag{77}$$

Introducimos estos datos en la ventana principal del programa:

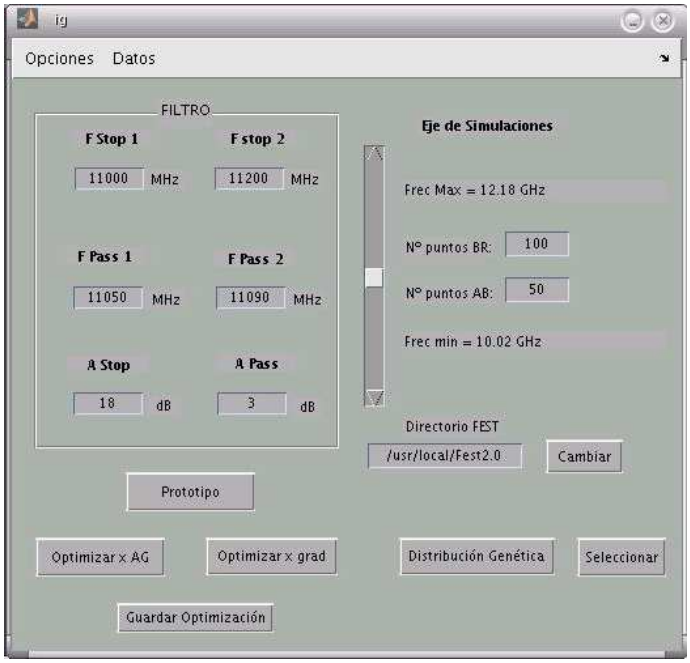

Figura 102. Ventana principal del programa

El prototipo teórico indica que se busca un filtro de orden 2.

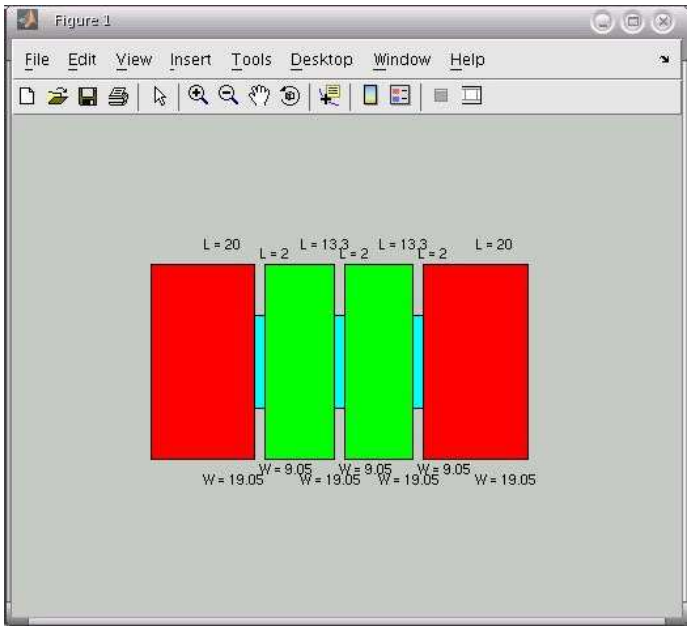

Figura 103. Dimensiones del filtro inicial

Vamos a emplear una población grande, de 200 individuos, para aumentar las posibilidades de encontrar buenos elementos, y la ruleta como modo de selección:

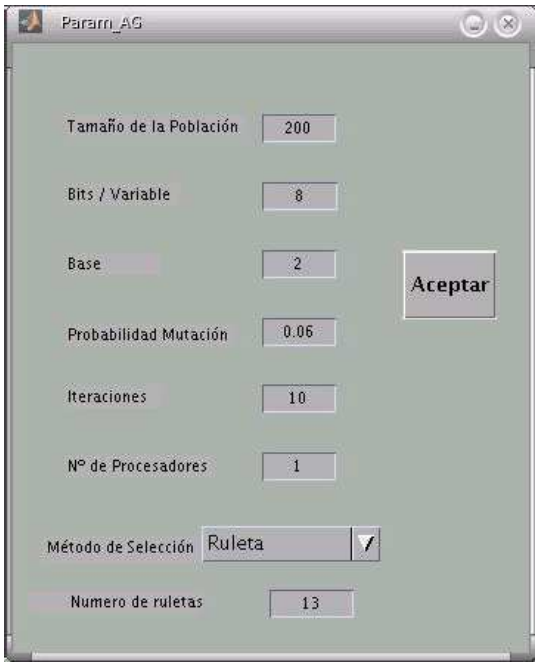

Figura 104. Parámetros del Algoritmo Genético

Al tratarse de una población grande con muchas ruletas, la riqueza genética está más asegurada, por lo que podemos iterar el algoritmo más veces que en otros ejemplos. Sin embargo, es preferible realizar sucesivos ciclos de pocas iteraciones en lugar de uno largo, para tener un mayor control durante el proceso. Las variables a optimizar son la longitud de las cavidades y la anchura de las discontinuidades, como en anteriores ejemplos:

Figura 105. Variables seleccionadas para optimizar

Nótese que el efecto de seleccionar la casilla *All* para la longitud de las cavidades resonantes en lugar de la correspondiente a la cavidad 1 es el mismo: al ser un filtro de 2º orden, si escogemos la casilla de la cavidad 1 asignará automáticamente el mismo valor a la cavidad 2 para mantener la simetría; la casilla *All* asigna la misma variable a todas las cavidades, en este caso, las dos que hay. Por este motivo, para

diferenciar la anchura de las discontinuidades de los extremos de la central, es necesario seleccionar las casillas 1 y 2 (la 3 tomará el valor de la 1 por simetría) en lugar de *All* para el caso del ancho de los escalones. Se intenta optimizar simultáneamente tanto el centrado de la banda de paso como el acoplo entre cavidades para evitar el problema de un excesivo rizado que coincida con la banda del filtro.

El algoritmo ha seguido la siguiente evolución:

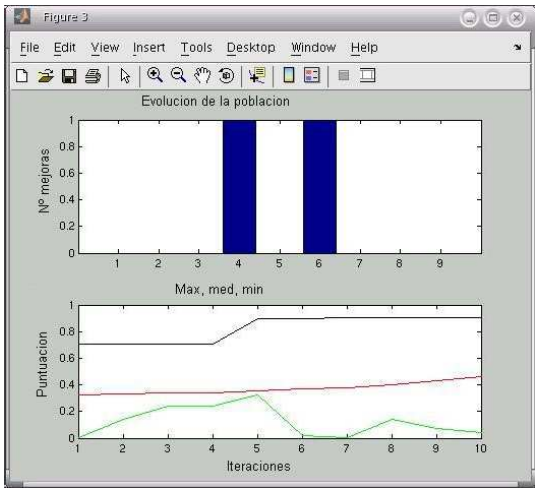

Figura 106. Evolución seguida por la población en 10 iteraciones

Puede verse cómo en las primeras iteraciones no se consigue ningún individuo mejor que los que se tienen, y que la puntuación mínima se acerca progresivamente a la media. Esto es señal de que las ruletas van eliminando los peores elementos sustituyéndolos por otros con puntuaciones medias; si hubiese muchos individuos con puntuación alta, se producirían más cruces favorables y la media subiría. Podemos deducir que en la población escasean estos últimos, habiendo menos que ruletas. En las iteraciones centrales surgen elementos más evolucionados aumentando la puntuación máxima y la media, a pesar de que también aparecen elementos con puntuaciones peores. Esto mismo puede apreciarse en la representación de las puntuaciones gracias al botón *Seleccionar*:

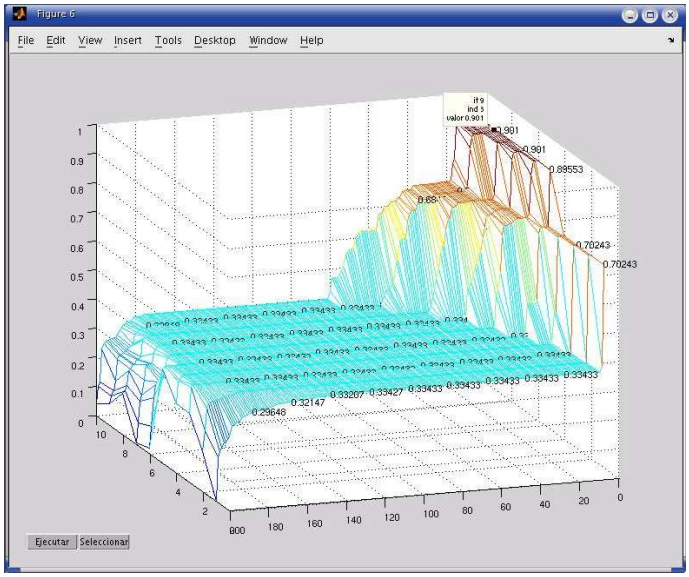

Figura 107. Selección de un individuo para crear una nueva población

A pesar de haber logrado algún individuo con puntuación superior a 0′9, el diseño puede mejorarse. Para ello se van a realizar 10 iteraciones más, generando una nueva población a partir de un filtro escogido de la gráfica anterior. En concreto, el filtro nº 3 de la iteración 9ª, que presenta una puntuación de 0′901. Se cambia el modo de selección al torneo simple:

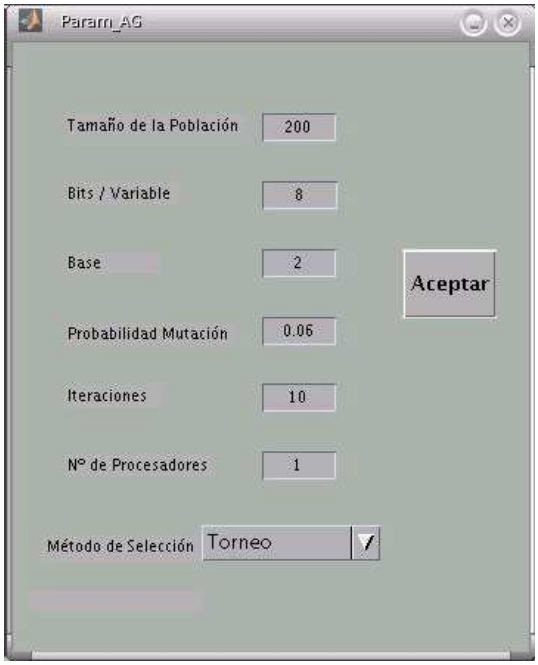

Figura 108. Parámetros de la nueva población

La evolución de la población en este caso no presenta grandes mejorías, pero hay que tener en cuenta que desde la primera generación se cuenta con individuos muy avanzados:

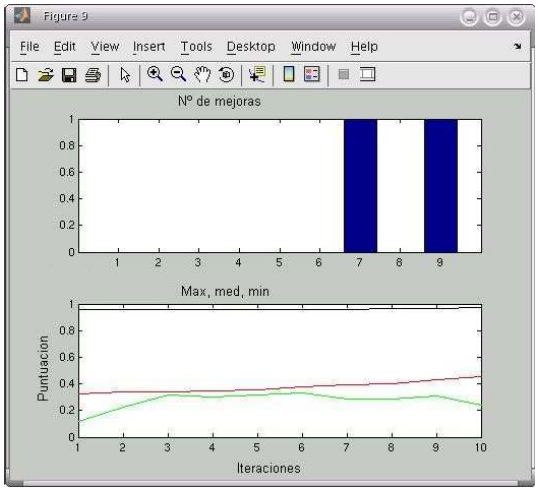

Figura 109. Evolución seguida por la nueva población

Para comprobar si es factible seguir iterando sobre la misma población nos fijamos en la distribución genética de sus variables en la última generación:

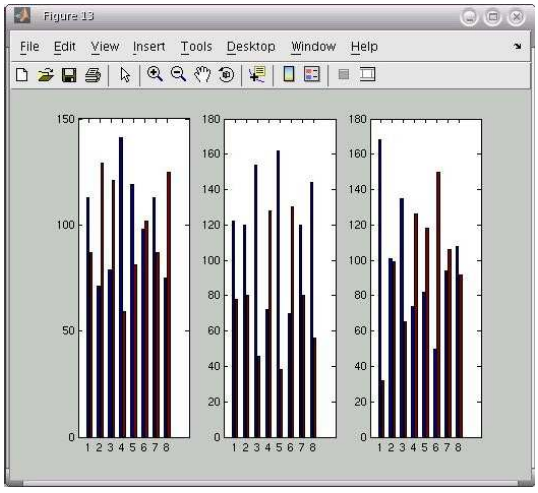

Figura 110. Distribución genética de la 10ª iteración

Esta población se encuentra bastante desgastada, sobre todo las variables 2 y 3. La superficie de puntuaciones es bastante similar a la anterior, lo cual es señal de la dificultad que tiene el algoritmo para encontrar muestras que cumplan las especificaciones requeridas:

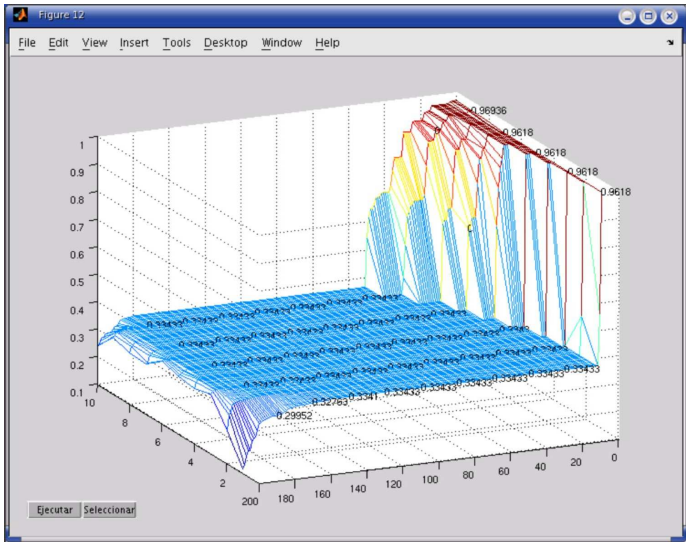

Figura 111. Puntuaciones obtenidas en la nueva población

Aún así, puede verse cómo el número de muestras positivas aumenta en cada iteración. La gráfica siguiente compara las respuestas de los filtros obtenidos hasta ahora:

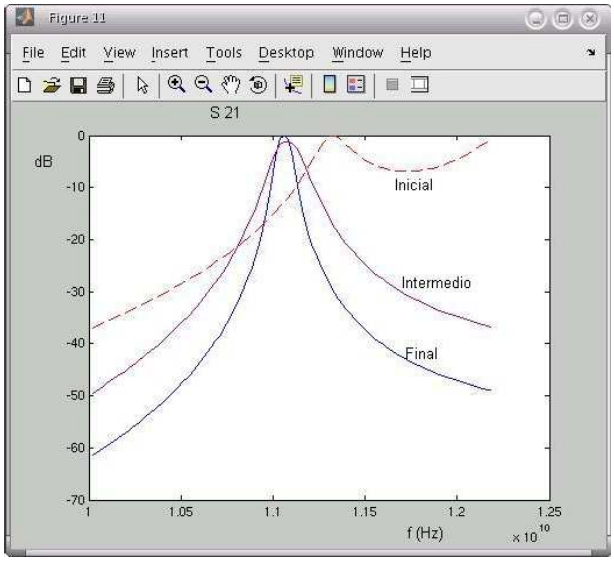

Figura 112. Comparación de las respuestas de los filtros conseguidos

La línea discontinua corresponde al filtro teórico con el que se creó la primera población; la roja es del filtro que sirvió de modelo para la segunda población; la azul es la respuesta del mejor filtro obtenido tras las últimas iteraciones. Vemos cómo el programa ha sido capaz de centrar y reducir la banda de paso hasta aproximarse al filtro deseado. Por último se realiza una optimización por gradiente para conseguir un filtro con un ancho de banda más ajustado al ideal. El resultado aparece reflejado en negro sobre la siguiente figura:

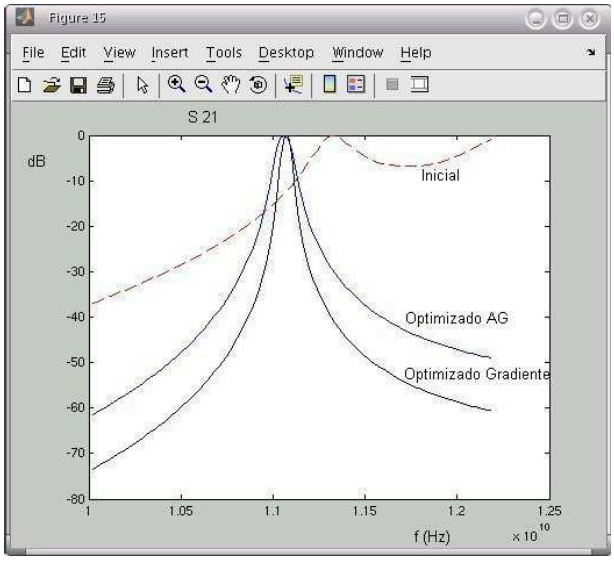

Figura 113. Respuesta tras optimizar por gradiente

La respuesta de este filtro se corresponde totalmente con las especificaciones del principio.

Comparemos las dimensiones de los filtros obtenidos durante el proceso. La primera representación corresponde al filtro alcanzado después de la primera optimización, 10 iteraciones sobre el modelo teórico:

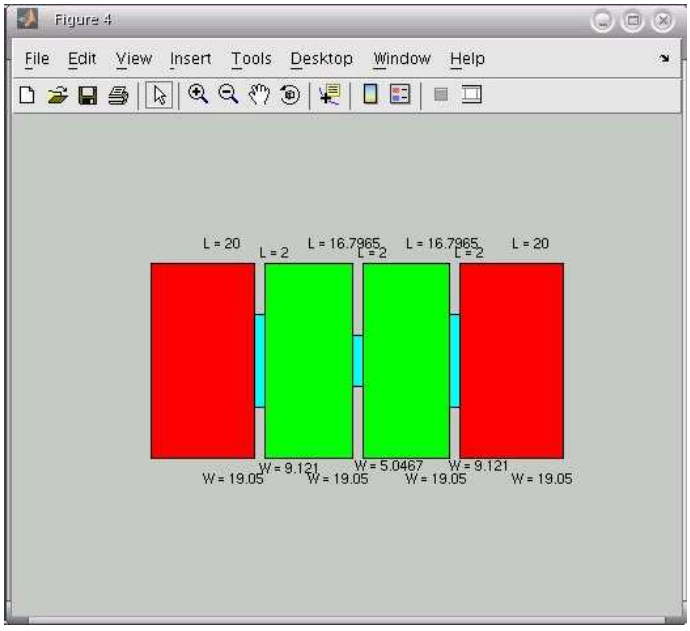

Figura 114. Dimensiones del filtro tras las 10 primeras iteraciones

Como se ve, aumenta la longitud de las cavidades para bajar la banda de paso de 11'75 GHz a 11 GHz; también se cierra la abertura entre ambas cavidades para mejorar su acoplo, disminuyendo el ancho de banda y su rizado. A partir de este filtro se crea la segunda población, que desemboca en el siguiente resultado:

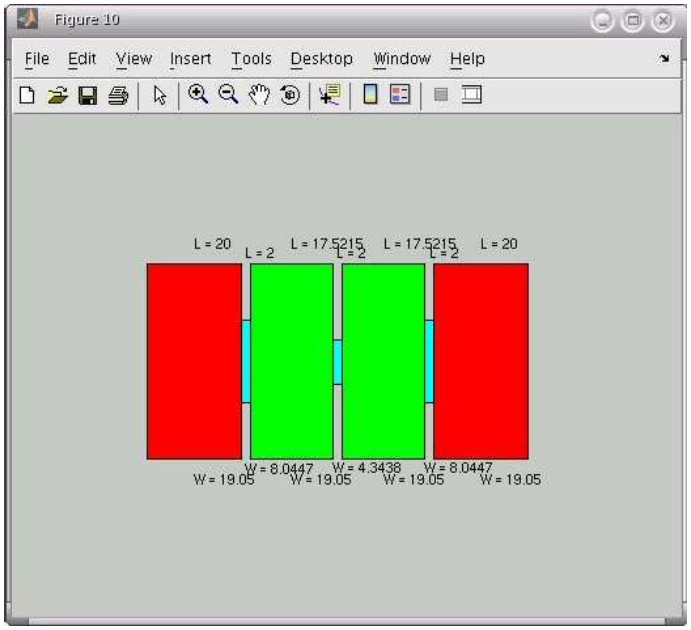

Figura 115. Dimensiones del filtro optimizado por A.G.

Este filtro continúa la tendencia de cerrar la discontinuidad central y aumentar la longitud de las cavidades, aunque en menor medida, dado que el ajuste principal ya está hecho. Las discontinuidades de los extremos se estrechan ligeramente, lo que combinado con la reducción del escalón central, permite reducir el rizado en la banda de paso. Después se realizó una optimización por gradiente, resultando el filtro finalmente optimizado:

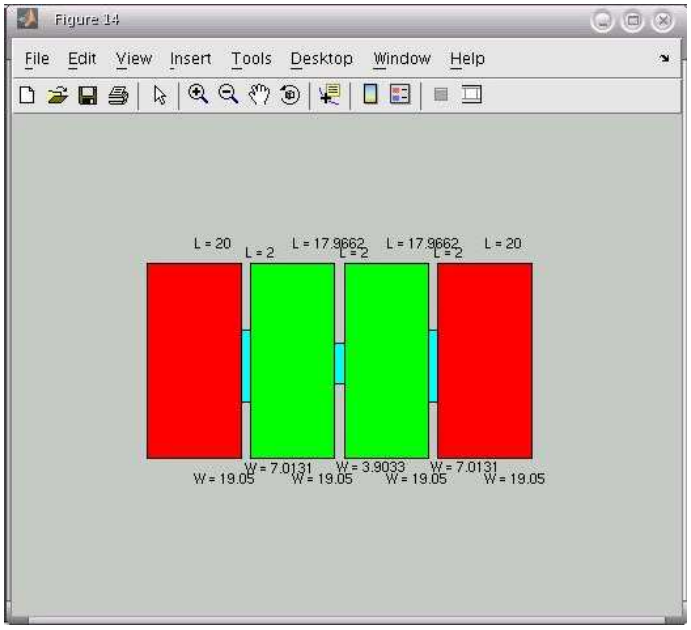

Figura 116. Dimensiones del filtro optimizado por gradiente

De nuevo se estrechan las discontinuidades, especialmente las laterales. A pesar de que los cambios de tamaño producidos en este último paso son pequeños, la respuesta del filtro mejoraba visiblemente, disminuyendo el rizado.

5. Anexo: Procesado en paralelo

El empleo de algoritmos genéticos requiere realizar un gran número de simulaciones, mucho mayor que con otras técnicas. Por ejemplo, para un proceso de 8 iteraciones sobre una población de 100 individuos se necesitan 800 simulaciones. Si un ordenador invierte 3 segundos en cada simulación, necesitará 40 minutos para completar la optimización. Para reducir este tiempo pueden utilizarse varios procesadores simultáneamente, repartiendo entre ellos el trabajo a realizar. En el caso anterior, si en vez de 1 procesador se utilizasen 2, el tiempo se reduciría a la mitad: 20 minutos.

Se designa un procesador como principal y el resto como secundarios. El primero se encarga de todas las tareas del programa (interfaz con el usuario, creación del filtro prototipo, población inicial, cruces, presentación de resultados, etc) excepto de las simulaciones y evaluaciones; estas las reparte entre todos los procesadores (secundarios y él mismo). Los procesadores secundarios permanecen inactivos el resto del tiempo.

Más detalladamente, el proceso de computación distribuida [4] se realiza del siguiente modo: el procesador principal guarda en un archivo los datos necesarios para las simulaciones (dimensiones de los filtros de la población y eje de frecuencias), así como el número de procesadores que se utilizarán. Una vez cerrado el archivo de datos pueden comenzar las simulaciones, por lo que despierta a los programas secundarios de los otros procesadores. Estos programas leen el archivo de datos y seleccionan una fracción de la población para trabajar sobre ella. Cada secundario tiene asignado un número, correlativos, que lo identifica y le indica qué parte de la población debe simular. Así, siguiendo con el ejemplo de arriba, el nº 2 se encargaría de los individuos 21 a 40 de entre los 100 a estudiar en cada iteración. Cuando terminan, guardan los resultados en un archivo común. Como el volumen de datos sería excesivo si se almacenasen las salidas de las simulaciones generadas por *FEST* (una matriz con los valores de los parámetros **S**, módulo y fase, para cada punto de frecuencia), el programa secundario debe encargarse también de la evaluación de los individuos y guardar en el archivo solamente la puntuación obtenida por cada uno de estos. Una vez hecho esto, el programa principal recoge los datos del arhivo de resultados y continúa el algoritmo genético; los programas secundarios entran en hibernación hasta que vuelven a ser requeridos en la siguiente iteración.

Para que los programas secundarios sepan cuándo deben leer el archivo de datos y comenzar un nuevo ciclo de simulaciones, permanecen en espera activa controlados por un contador accesible a todos. Este contador se encuentra inicialmente a 0; cuando el programa principal reescribe el archivo de entrada lo pone a 1, señal que esperan los secundarios para arrancar. Cuando uno termina, espera hasta que el contador iguale su identificador; entonces escribe sus evaluaciones en el archivo de resultados e incrementa el contador en una unidad. El programa principal, tras realizar sus correspondientes simulaciones, espera hasta que el contador iguale el número de procesadores (señal de que todos han acabado); entonces lo pone a 0 y carga los resultados para seguir su proceso. La figura siguiente ilustra el comportamiento de los porgramas principal y secundarios, y la comunicación que mantienen mediante el manejo del archivo contador:

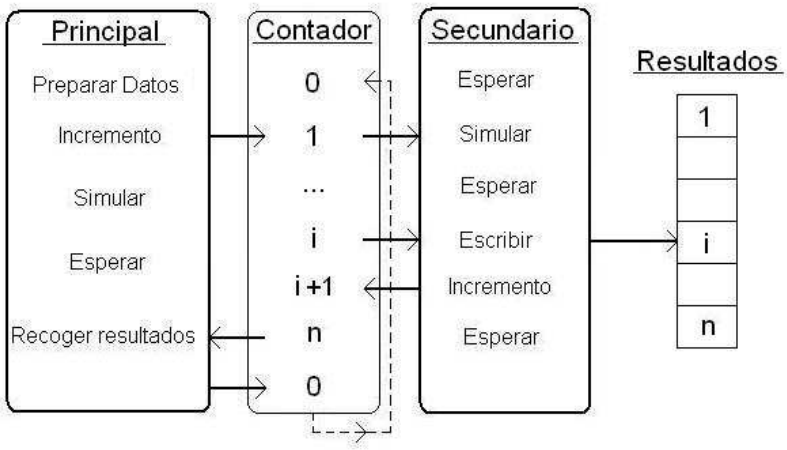

Figura 117. Diagrama de transferencia de datos entre el programa principal y los secundarios

Dos programas no pueden acceder al archivo de resultados al mismo tiempo, porque lo hacen para escribir y sólo se guardarían los cambios introducidos por el último en salir. Por eso se ha decidido que el acceso sea por turnos, controlado por el contador. Aunque un procesador más rápido tenga que esperar para escribir sus resultados, el tiempo global del proceso no aumenta; este permanece fijado por el procesador más lento, cuyos resultados se necesitan antes de continuar con otra iteración.

Todos los ejemplos mostrados en el capítulo anterior se han realizado sin el uso de procesadores secundarios. Esta herramienta se encuentra totalmente desarrollada, excepto el acceso automático (sin contraseña manual) de un PC a otro necesario en cada iteración del proceso.

Referencias

[1] Bishop, Crhistopher."Neural networks for pattern recognition". Oxford University Press, 2003

[2] Rahmat-Samii, Yahya & Michielssen, Eric..Electromagnetic optimization by genetic algorithms". John Wiley & Son, 1999

[3] Hunter,Ian."Theory and design of microwave filters". Instituion of Electrical Engineers, 2001

[4] Pérez Martínez, Jorge."Programación concurrente". Ed. Rueda, 1990