UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

Síntesis dimensional de filtros de microondas mediante la técnica del slope parameter

Damián Pla Herliczka

Tutor: Pablo Soto Pacheco

Cotutor: José Vicente Morro Ros

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2019-20

Valencia, 6 de Septiembre de 2020

Resumen

El presente Trabajo de Fin de Grado recoge la implementación en *MATLAB* de una mejora en el método clásico de diseño de filtros de microondas basado en un prototipo circuital equivalente. Dicha mejora permite tener en cuenta la dependencia en frecuencia de los elementos que forman las estructuras de microondas y las interacciones entre ellos. Esto se consigue añadiendo al *slope parameter* los efectos de los elementos de acoplo adyacentes y después utilizarlo como base para sintetizar los resonadores de los filtros. De este modo, es posible acercar la equivalencia entre el prototipo paso banda y la estructura, resultando en la capacidad de sintetizar estructuras de alta calidad en pocas iteraciones y la disminución del coste computacional asociado a la simplificación o incluso eliminación de la optimización final. El objetivo de este TFG será la implementación y automatización del método en un script de *MATLAB* que utilice *FEST3D* para diseñar filtros de modo evanescente compuestos de guías *ridge* y *rectangulares*.

Resum

El present Treball de Fi de Grau recull la implementació a *MATLAB* d'una millora al mètode clàsic de disseny de filtres de microones basat en el prototip circuital equivalent. Aquesta millora permet tindre en compte la dependència en freqüència dels elements que formen les estructures de microones i les interaccions entre ells. Açò s'acosegueix afegint al *slope parameter* els efectes dels elements d'acoblament adjacents i després utilitzant-lo com a base per a sintetitzar els resonadors dels filtres. D'aquesta manera és possible apropar l'equivalència entre el prototip pas banda i l'estructura obtinguda, resultant en la capacitat de sintetitzar estructures d'alta qualitat en poques iteracions i la disminució del cost computacional associat a la simplificació i fins i tot l'eliminació de la optimització final. L'objectiu d'aquest TFG serà la implementació i automatització del mètode a un script de *MATLAB* que utilitze *FEST3D* per al disseny de filtres de mode evanescent compostos de guies *ridge* i *rectangulars*.

Abstract

This Bachelor dissertation includes the implementation in *MATLAB* of an improvement in the classic microwave filter design method based on an equivalent circuit prototype. This improvement makes it possible to take into account the frequency dependence of the elements that form the microwave structures and their interactions. This is achieved by adding the effects of the adjacent coupling elements to the slope parameter and then using it as a basis for synthesizing the filter resonators. That way, it is possible to improve the equivalence between the band-pass prototype and the obtained structure, resulting in the ability to synthesize high-quality microwave structures in a few iterations and in a reduction of the computational cost due to the simplification or even elimination of the final optimization step. The goal of this dissertation will be the implementation and automation of the method in a *MATLAB* script that uses *FEST3D* to design evanescent mode filters composed of *ridge* and *rectangular* waveguides.

Índice general

II Anexos

Índice de figuras

Índice de tablas

Parte I

Memoria

Capítulo 1

Introducción

1.1. Estado del arte

1.1.1. Sistemas de comunicación

La comunicación es el proceso por el cual se transforma y transmite cierta información entre dos puntos físicamente separados. Durante la historia este proceso ha experimentado numerosos cambios, desde el uso de palomas mensajeras o señales de humo, hasta los complejos y sofisticados sistemas actuales, como los satélites de comunicaciones o las comunicaciones por fibra óptica.

Los primeros sistemas de comunicaciones eléctricos, además de presentar una baja fiabilidad (fruto de la variabilidad del medio y la incapacidad en aquel entonces de prevenir errores), estaban estrechamente ligados a la distancia y la cantidad de información que se quería transmitir. Con el paso del tiempo, y gracias a los continuos avances en el conocimiento científico, se crearon nuevos sistemas que permitían comunicaciones con un volumen datos y velocidades de transmisión superiores, aumentando la eficiencia hasta situarse muy cerca del límite teórico de Shannon. La aparición de estos sistemas ha cambiado por completo la sociedad y nos ha introducido de pleno en el momento actual, la Era de las Comunicaciones (también conocida como Era Digital o de la Sociedad de la Información).

Figura 1.1: Diagrama simplificado de un sistema de comunicación

El sistema de comunicación mas básico se puede resumir en el diagrama de bloques de la Figura 1.1, que representa las diferentes transformaciones y fases que atraviesa la información desde su fuente hasta su destino. Aunque este simple diagrama puede ser valido para cualquier sistema de comunicación, en los distintos ejemplos nos referiremos siempre al caso de un sistema de comunicación de radiofrecuencia.

En tales sistemas el canal vendrá dado por espacio libre (ya sean las diferentes capas de la atmósfera o el espacio exterior que atraviese la señal), al que el bloque *Transmisor* y *Receptor* accederán a través de antenas. La información, habiendo surgido de la fuente, sera enviada al bloque

Transmisor, que se encargara de realizar las tareas u operaciones tales como la cuantificación, encriptación, codificación, y modulación... adecuando la señal lo máximo posible para su posterior transmisión, y luego de acceder a la antena emisora que la enviará por el *canal* radio. Este canal añadirá cierto ruido e interferencias a nuestra señal, la atenuará, y cuando llegue finalmente al bloque *Receptor*, éste se encargará de realizar las operaciones inversas al transmisor y corregirá los errores que se hayan podido producir. Finalmente, la información captada deberá ser una réplica suficientemente buena de la enviada, se entregará al bloque *Destino*.

1.1.2. Transmisor y receptor

Como hemos visto en la Subsección 1.1.1, todo sistema de comunicaciones, por una parte, cuenta con un transmisor que se encarga de hacer las transformaciones necesarias a la información, favoreciendo su transmisión por el canal en las mejores condiciones posibles, y por otra parte, un receptor que realiza las funciones contrarias para recuperar la información original a partir de la señal recibida por el canal radio. En nuestro caso nos centraremos en el bloque receptor ya que para nuestro trabajo representa un ejemplo más ilustrativo. Un receptor típico se puede desglosar en el diagrama de bloques de la Figura 1.2 [1].

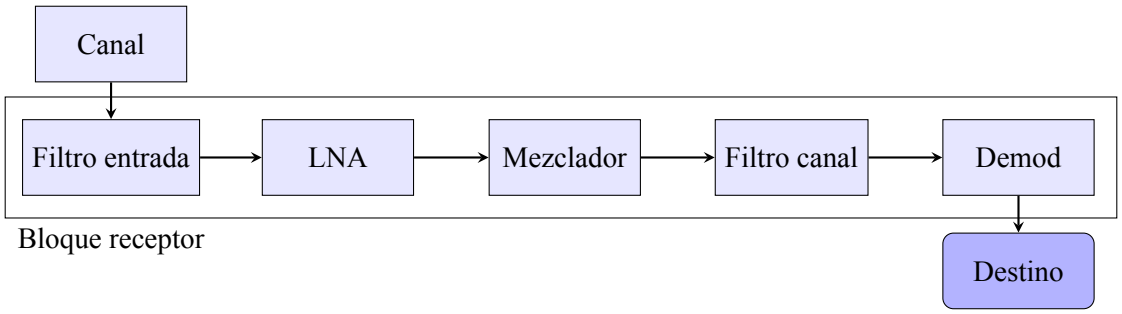


Figura 1.2: Diagrama de bloques equivalente al bloque receptor

De estos bloques:

- *Filtro entrada*: es el filtro preselector que se encarga de eliminar las bandas no deseadas a la entrada del sistema, así como las frecuencias imagen asociadas a la posterior conversión en frecuencia. De esta forma se elimina ruido e interferencias de otras bandas que puedan mezclarse en el receptor con la señal que queremos captar.

- *LNA*: es un amplificador de bajo ruido, que permite amplificar señales débiles sin aportarle un ruido alto, logrando así que la señal sea más robusta al ruido del resto de elementos del receptor. Es un elemento esencial para aumentar la sensibilidad de cualquier receptor.

- *Mezclador*: este bloque realiza un batido de la señal a su entrada con la de un oscilador local interno, originando una conversión en frecuencia de dicha señal a la entrada. El oscilador local se sintoniza para lograr que el canal concreto que se pretende captar se ubique a la salida del mezclador a la frecuencia intermedia del receptor.

- *Filtro canal*: se trata de un filtro con el ancho de banda del canal y centrado a la frecuencia intermedia del receptor, que se encarga de dejar pasar sólo al canal deseado y eliminar el resto de canales de la banda seleccionada por el filtro preselector. Por tanto, se trata del filtro de selección del canal.

- *Demod*: es el demodulador de la señal, que la baja a banda base y realiza las funciones de decodificación (que incluye la corrección de errores) para recuperar la información original.

En este proyecto nos centraremos en los bloques encargados de eliminar las frecuencias no deseadas, es decir, los filtros, elementos imprescindibles en cualquier sistema de comunicaciones. En recepción, como hemos visto, se encargan de seleccionar la banda y el canal deseado, mientras que en transmisión su función consiste en limpiar la señal (eliminando componentes espectrales indeseadas que pueden originar interferencias con otros sistemas) antes de emitirla.

1.1.3. Filtros en comunicaciones

Gracias al trabajo que realizaron científicos como Ampère, Ohm, Faraday, Coulomb, Gauss o Maxwell, se fundaron las bases del electromagnetismo. A partir de ellas comenzó el desarrollo de las comunicaciones hace mas de 150 años, y que nos han llevado a la maravilla tecnológica de la que hoy disponemos.

A principios del siglo pasado, coincidiendo con los pasos iniciales en el desarrollo de los primeros y rudimentarios sistemas de comunicaciones por radiofrecuencia, se observó la necesidad de disponer de filtros que dejaran pasar la señal ubicada en cierta parte del espectro y rechazar el resto. De dicha época cabe destacar el método de los *Parámetros Imagen*, creado por Zobel en 1923, que permite diseñar filtros periódicos a partir del conocimiento de las frecuencias de corte y rechazos requeridos, y que está basado en explotar las bandas de paso y prohibidas asociadas a las estructuras periódicas (retomadas recientemente, por ejemplo, en la nueva disciplina de los metamateriales). La Segunda Guerra Mundial aumentó la importancia estratégica de los sistemas de comunicaciones y el radar, produciéndose un importante avance en estas áreas. Dentro de este marco histórico, se produjo también una significativa mejoría en la Teoría de Filtros, puesto que se establecieron las bases para su diseño sistemático gracias al trabajo realizado por Darlington y Cauer para la síntesis de filtros por el *Método de las Pérdidas de Inserción* [2] [3]. Este método permitía tener un control total de la respuesta (a diferencia del método de los *Parámetros Imagen*), mejorando los procedimiento basados en prueba y error utilizados hasta la fecha. Aunque la mayor parte de este método fue publicado en los años 50, ya se utilizaba desde la década de los 30, sobre todo en el ámbito militar.

Otro de los puntos de inflexión en el ámbito del diseño de filtros está asociado al artículo que publicó Cohn en 1957 [4]. Este trabajo propone la síntesis de un prototipo formado por inversores de inmitancias y resonadores, que permite establecer una correspondencia más precisa y simple con los elementos de las estructuras reales de microondas. Gran parte de los métodos de diseño de filtros de microondas actuales están basados en esta técnica [1], [5]. Así mismo, en el artículo de Cohn aparece el concepto del *slope parameter*, aunque sin todavía llegar a definirlo como tal.

En todas las aplicaciones de las telecomunicaciones es necesario el uso de filtros que eliminen señales y bandas no deseadas del espectro electromagnético y, por lo general, intentando disminuir las pérdidas que este proceso pueda introducir. Con este fin, en este documento se trabajará con

filtros que operan en la banda de microondas (que según su definición cubre el margen de frecuencias entre 300 MHz y 300 GHz, si bien a nivel de comunicaciones vía radio suele hacer referencia a la banda entre 1 GHz y 100 GHz en la que operan dichos sistemas).

El uso de bandas altas de frecuencias produce tanto beneficios como problemas a la hora de diseñar los elementos. Por una parte, nos permite tener anchos de banda absolutos mayores (lo que se traduce en mayores capacidades de transmisión) y es posible disminuir el tamaño físico de los dispositivos. Esto resulta beneficioso ya que, por ejemplo, en comunicaciones espaciales el tamaño y peso de los dispositivos es un factor muy importante a tener en cuenta. No obstante, por otra parte, con el aumento de la frecuencia también viene asociado un aumento en la absorción atmosférica [1], así como problemas asociados a la precisión de fabricación (dado que al disminuir las dimensiones de los elementos, las imperfecciones producen efectos indeseados más notables). Debido a los exigentes requerimientos a nivel de potencia y de pérdidas en las aplicaciones espaciales, la mayoría de estos filtros son diseñados en tecnología de guía de ondas. El uso de altas potencias a frecuencia elevadas (donde las dimensiones son reducidas, debido a una menor longitud de onda) contribuyen a la aparición de efectos indeseados como el *multipactor*, *corona* o la *intermodulación pasiva* que deben ser mantenidas bajo control [6].

Respecto al diseño de los filtros de microondas, el procedimiento clásico de diseño a día de hoy aún esta basado en el método de Cohn [4]. Este procedimiento sigue una serie de pasos:

1. Síntesis de un modelo circuital del filtro con la simplicidad suficiente para conseguir una solución analítica.

2. Síntesis inicial de la estructura real. Esto se consigue relacionando el modelo circuital y las partes de la estructura real, con lo que se conseguirán una dimensiones aproximadas del filtro.

3. Optimizar las dimensiones iniciales que se han obtenido en el segundo paso para obtener una dispositivo final con una respuesta lo más parecida posible a la deseada.

El mayor problema de esta técnica reside en la similitud entre el modelo circuital equivalente sintetizado de forma analítica en el paso 1 y la estructura real. Cuando el modelo circuital es una buena representación de la realidad, es posible extraer unas excelentes dimensiones iniciales en el paso 2, que simplifican la optimización final. En caso contrario, sin embargo, las dimensiones obtenidas tras la síntesis dimensional de la estructura no son buenas, requiriendo un elevado esfuerzo de optimización hasta obtener una respuesta que cumpla las especificaciones planteadas.

Así por ejemplo, el método tradicional de diseño suele ser válido para anchos de banda relativos estrechos, funcionando mejor cuanto menor sea el ancho de banda. No obstante, cuando el ancho de banda relativo empieza a ser superior al 3 % [7], las interacciones entre los elementos reales de la estructura y la variación en frecuencia de dichos elementos empiezan a producir cambios en la respuesta que no son tenidos en cuenta por el modelo circuital, dando lugar a unas peores dimensiones iniciales. Para estos casos es necesario desarrollar modelos que representen de una forma más precisa a la estructura real de filtrado.

La técnica de Cohn, y la mayor parte de las utilizadas actualmente, toman el *slope parameter* de un resonador ideal aislado. Sin embargo, la presencia de los elementos de acoplo a la entrada y a la salida de cada resonador afectan a su comportamiento. Recientemente, se ha propuesto una técnica que propone modificar el *slope parameter* del resonador incluyendo el efecto de dichos elementos

de acoplo [8], y que ha obtenido buenos resultados para ciertas topologías concretas. Este método ha sido aplicado, con mayor o menor éxito, a filtros inductivos en guía de onda rectangular [9] y filtros de modos evanescente [10]. No obstante, en estas técnicas se analiza la ventana de acoplo y el resonador de forma separada. En el trabajo a realizar en el presente Trabajo Final de Grado se pretende mejorar este procedimiento analizando de forma integrada el efecto del resonador y las ventanas de acoplo, lo que permite obtener de forma precisa el *slope parameter* del resonador en las condiciones de carga reales a las que está sometido y teniendo presente su variación con la frecuencia. Se espera por tanto lograr una mayor precisión en la síntesis dimensional, que podría permitir abordar el diseño de filtros de anchos de banda moderados (en el rango entre un 3 % y un 10 % de ancho de banda relativo) o incluso hacer innecesaria la realización de una optimización final (o al menos, a que consista en un pequeño reajuste final en las dimensiones de la estructura).

1.1.4. Modelo ideal de filtro y sus características principales

Los filtros, como bien hemos dicho, son elementos dentro de los sistemas de comunicaciones que se encargan de eliminar las bandas de frecuencia que no se requieren. Existen una serie de características principales y propiedades que determinan su respuesta (ver Figura 1.3). Entre ellas podemos citar:

- **Banda de paso:** es la banda de frecuencias en la que el filtro permite el paso de la señal, y donde se intenta que ésta sufra la menor distorsión posible. Se suele indicar con la frecuencia de corte inferior y superior de dicha banda de paso, f_1 y f_2 respectivamente.

- **Ancho de banda:** se trata del margen de frecuencias que abarca la banda de paso, es decir, $f_2 - f_1$, y suele indicarse como BW o Δf.

- **Frecuencia central:** es el parámetro que indica la frecuencia central del filtro y se denota por f_0. Normalmente se calcula como la media geométrica de las frecuencias de corte de la banda de paso, es decir, $f_0 = \sqrt{f_1 f_2}$.

- **Ancho de banda relativo:** este parámetro indica el ancho de banda respecto a la frecuencia central (normalmente se da en porcentaje). Se indica como $BW_\mathrm{r} = \Delta = \frac{\Delta f}{f_0}$.

- **Banda atenuada o eliminada:** parte del espectro que no se desea a la salida del filtro. Un filtro tendría que presentar idealmente una atenuación infinita en la banda eliminada. En la práctica nos conformamos con que la atenuación o rechazo sea superior a un cierto nivel mínimo necesario para la aplicación (típicamente 40 o 60 dB).

- **Banda de transición:** puesto que en la práctica es imposible implementar un filtro ideal que pase de una atenuación nula a infinita de forma inmediata, se suele tener una banda donde se realiza una transición entre la banda de paso y la banda atenuada, y en la que la atenuación va gradualmente aumentando.

- **Pérdidas de inserción en la banda de paso:** representa la atenuación que sufrirá una señal ubicada en la banda de paso del filtro al atravesarlo. Idealmente debería ser cero, si bien en la práctica siempre habrán pérdidas por las conductividad finita de los conductores, el aislamiento imperfecto de los dieléctricos, y las reflexiones (o pérdidas de retorno).

- **Pérdidas de retorno en la banda de paso:** hace referencia a que cantidad de la señal en la banda de paso es reflejada hacia atrás por el filtro, de modo que un valor mayor de este parámetro indica que la reflexión es menor. Idealmente deberían ser infinitas, pero no es conseguible en la práctica (de hecho, un valor más alto de las pérdidas de retorno trae emparejado un aumento de la banda de transición). Los valores típicos suelen estar en el rango entre 18 y 30 dB para tecnología guiada.

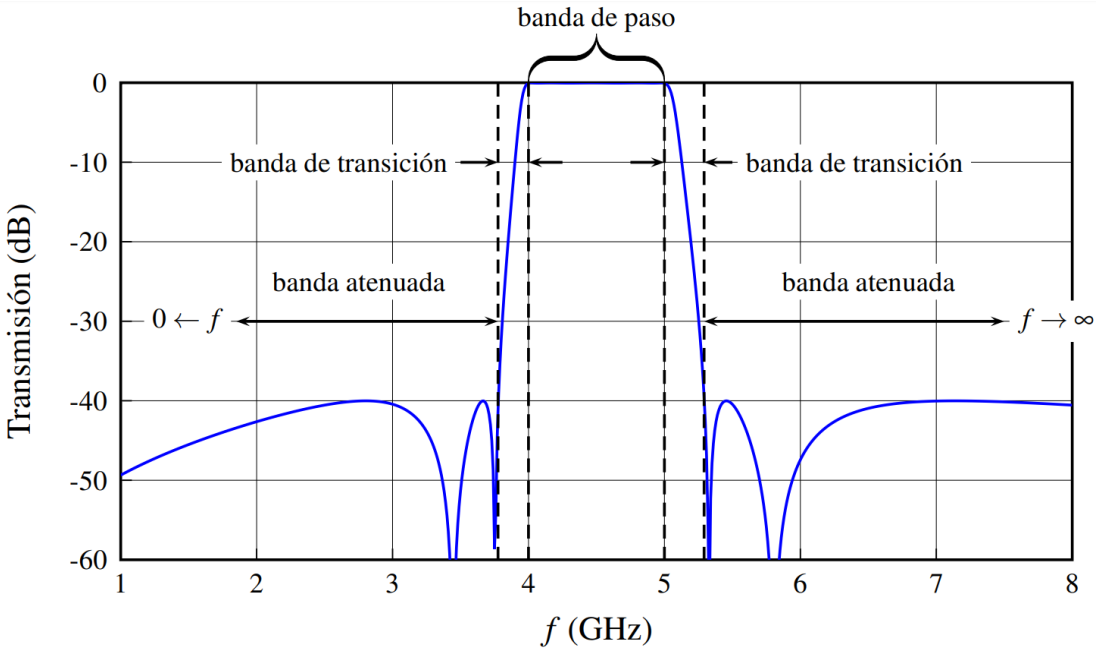


Figura 1.3: Filtro paso banda con banda de paso entre 4 GHz **y** 5 GHz **[11]**

Los filtros se implementan en la práctica mediante la agrupación de diferentes resonadores acoplados entre sí a través de aperturas (denominados elementos de acoplo). Al número de resonadores que lo componen se les denomina orden del filtro. Normalmente la respuesta puede tener tantos ceros de reflexión en el eje de frecuencias (es decir, frecuencias a las que la potencia a la entrada del filtro lo atraviesa sin reflexiones) como orden tenga el filtro. Un orden más elevado implica una respuesta más selectiva, pero a cambio tiene varios inconvenientes: aumentan las pérdidas, el tamaño, el peso y el volumen, y además la estructura es más sensible a errores de fabricación. Así mismo, órdenes más altos implican incrementar el tiempo de diseño y fabricación, con el consiguiente aumento de coste. Por dicho motivo, se intenta que los filtros tengan el orden más reducido que permitan lograr la selectividad (o rechazo en la banda eliminada) deseada.

A parte de la clasificación basada en el método de diseño o la tecnología de construcción, los filtros se pueden clasificar segun la respuesta en frecuencia que presentan:

- **Filtro paso banda:** permiten el paso de una banda central, con los límites f_1 y f_2, y atenuan las bandas laterales en las que $f < f_1$ y $f > f_2$. Obsérvese el ejemplo mostrado en Figura 1.3.
- **Filtro paso bajo:** permite el paso desde 0 Hz hasta una frecuencia de corte determinada f_c (es como si $f_1 = 0$ y $f_2 = f_c$). Para frecuencias superiores a la frecuecia de corte de la

banda de paso, la señal se atenua, idealmente de forma infinita. Obsérvese como ejemplo la Figura 1.4.

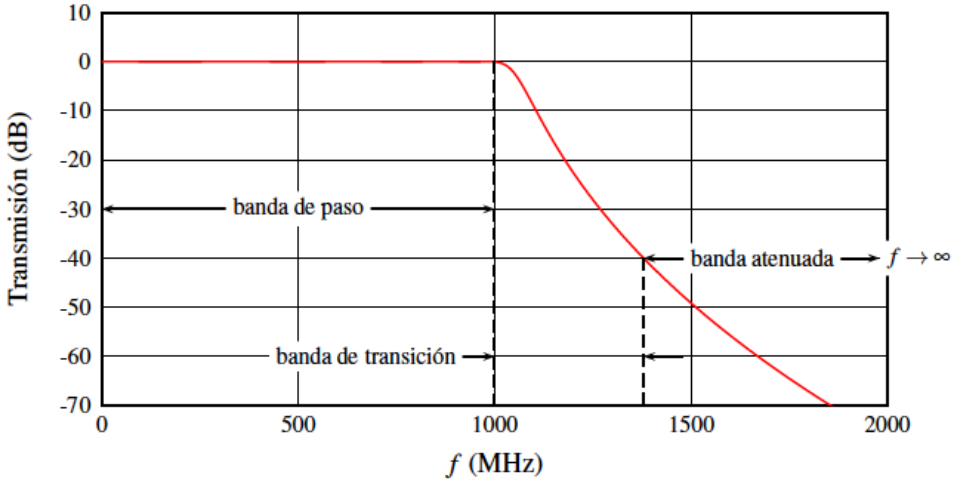


Figura 1.4: Filtro paso bajo con frecuencia de corte $f_c = 1$ GHz [11].

- **Filtro paso alto:** atenuan toda frecuencia situada entre 0 Hz y f_c. Una vez superada esta frecuencia la señal pasa, idealmente sin atenuación, hasta frecuencia infinita. Obsérvese el ejemplo de Figura 1.5.

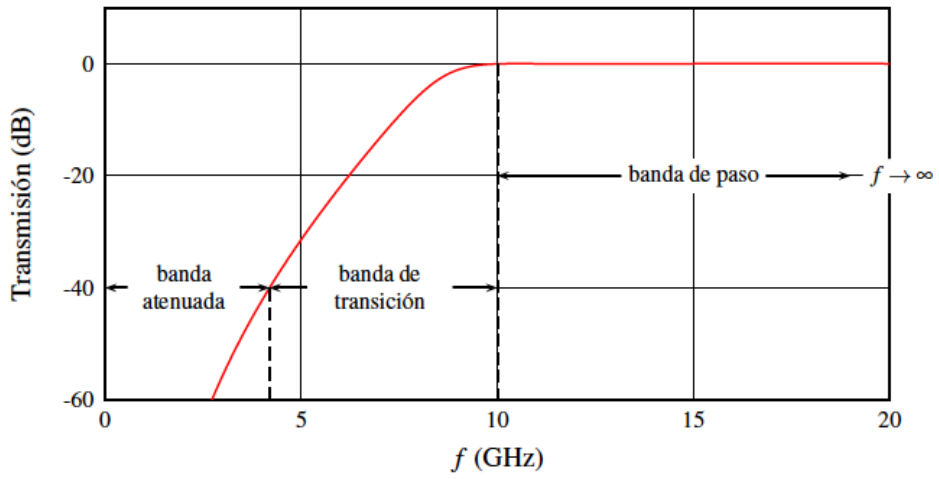


Figura 1.5: Filtro paso alto con frecuencia de corte $f_c = 10$ GHz [11].

- **Filtro de banda eliminada:** es el inverso al filtro paso banda. Este filtro unicamente elimina el paso de cierta banda de frecuencias, comprendia entre f_1 y f_2 y permite el paso del resto de frecuencias. Se muestra un ejemplo en la Figura 1.6.

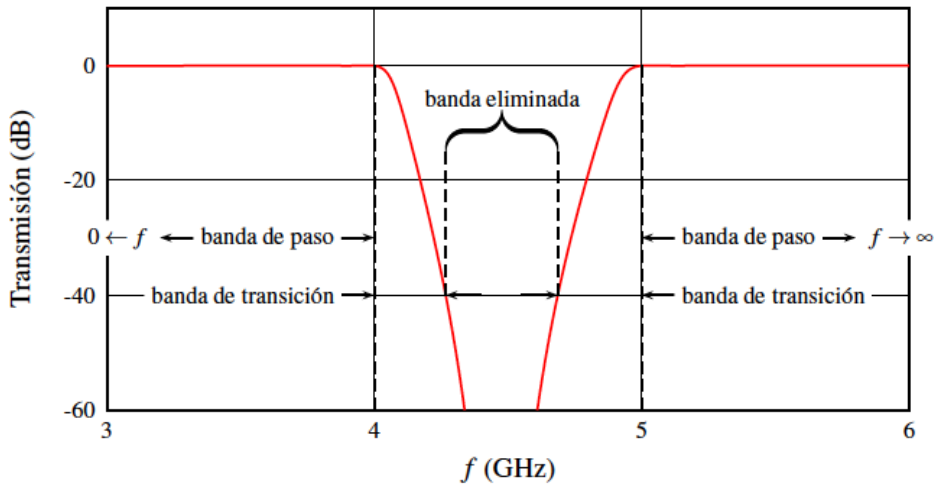


Figura 1.6: Filtro de banda eliminada con frecuencias $f_1 = 4\,\text{GHz}$ y $f_2 = 5\,\text{GHz}$ [11].

- **Filtro pasa todo:** estos filtros no eliminan ninguna banda de la señal, sino que se limitan a producir algun cambio en su fase en función de la frecuencia. Se trata por tanto de ecualizadores de fase.

Para las aplicaciones de comunicaciones, los filtros más habitualmente empleados son los paso banda. No obstante, los filtros paso bajo son siempre relevantes al ser los que se utilizan como punto de partida en las técnicas de síntesis. Por lo tanto, serán los dos tipos de filtros que trataremos en el presente Trabajo Final de Grado. Los filtros paso bajo darán las ecuaciones de partida del procedimiento teórico que se describirá en el Capítulo 2, a partir de las cuales se desarrollará un nuevo método para el diseño de filtros paso banda, basado en el *slope parameter*, y su implementación en tecnología en guía de onda.

1.2. Objetivo y metodología

1.2.1. Objetivo

El objetivo fundamental de este trabajo consiste en desarrollar un nuevo algoritmo que permita obtener unas excelentes dimensiones iniciales de filtros de microondas sin tener que recurrir a una costosa optimización posterior. Se pretende además que dicho método propsorcione buenos resultados incluso para anchos de banda moderados (del orden del 3 % al 10 % de ancho de banda relativo).

Para cumplir dicho objetivo general, se plantean los siguientes objetivos específicos:

- Desarrollo de un método de síntesis dimensional de filtros de microondas mediante la técnica del slope parameter
- Implementación de una aplicación software que realice dicha síntesis de forma automatizada
- Aplicación a un tipo concreto de estructuras de microondas.

La estructura a la que se aplicará inicialmente el método serán los filtros de modo evanescente en guía rectangular, al poder analizarse de forma precisa y eficiente mediante técnicas modeales.

1.2.2. Metodología

Para poder cumplir con los objetivos mencionados anteriormente en éste trabajo de investigación se ha seguido una metodología desglosable en 3 bloques o secciones principales:

- **Bloque 1:** Investigación del estado del arte en la teoría de filtros.
  - Aprendizaje de la base teórica y funcionamiento del método del *slope parameter*.
  - Búsqueda de información y lectura de artículos de investigación relacionados.
  - Revisión del estado del arte.
- **Bloque 2:** Diseño de ficheros de extensión ".fest" con *MATLAB*.
  - Aprendizaje del funcionamiento del software *FEST3D*.
  - Desarrollo de código en *MATLAB* de los bloques que representan las guías de onda que forman los filtros.
  - Definir las características físicas de los filtros a diseñar.
- **Bloque 3:** Implementación en el *script* principal de *MATLAB* encargado de realizar la síntesis automatizada de los filtros con el método del *slope parameter*.
  - Correcciones hasta conseguir funcionamiento deseado.
- **Bloque 4:** Diseño de filtros en guia de onda.
  - Aplicación del *script* de diseño de filtros basado en el *slope parameter* a estructuras de microondas.

- Comparación con otros métodos de diseño.
- Discusión de resultados.

Se pueden visualizar las relaciones y transiciones entre estos cuatro bloques principales mediante el diagrama de la Figura 1.7.

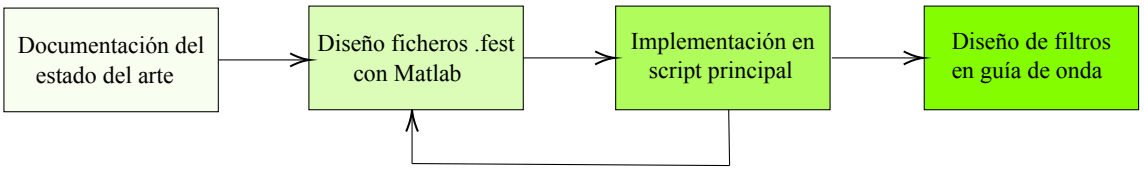


Figura 1.7: Diagrama de bloques de la metodología y sus relaciones

Los principales programas de software utilizados son *MATLAB* y *FEST3D*, este último integrado en el software *CST Studio Suite*. *FEST3D* es un software capaz de analizar complejas estructuras de microondas mediante la combinación de diferentes técnicas de análisis modal, que además incluye métodos numéricos para obtener los modos de guías y cavidades de forma arbitraria (como el método de BI-RME, del inglés *Boundary Integral-Resonant Mode Expansion*, que combina una expansión modal con una técnica de integral de contorno). Con *MATLAB* se han creado una serie de programas que crean los diferentes bloques de código que forman los archivos de simulación de *FEST3D*. Este ”creador de archivos” de extensión ”.fest” ha sido integrado posteriormente en un *script* principal que implementa el nuevo método de diseño basado en el *slope parameter*. Este *script* recibe como entrada los parámetros de la respuesta a obtener y las dimensiones fijas de la estructura a diseñar, y tras realizar la simulación y optimización usando la CLI de la que dispone *FEST3D*, devuelve las dimensiones físicas óptimas del filtro proporcionadas por el nuevo método.

En la Tabla 1.1 se puede ver la distribución temporal del trabajo de los bloques, con comienzo en febrero de 2020 y finalización en agosto de 2020:

Mes	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre
Bloque 1								
Bloque 2								
Bloque 3								
Bloque 4								

Tabla 1.1: Diagrama con la repartición temporal del trabajo de los diferentes bloques

1.3. Organización de la memoria

En el primer capítulo de esta memoria se hace una breve descripción sobre el estado del arte para situar al lector, para acto seguido formular los objetivos del proyecto y explicar la metodología que se ha seguido para conseguir dichos objetivos.

En el segundo capítulo se describirá en detalle un nuevo método de síntesis de filtros de microondas basado en la técnica del *slope parameter*. Se trata de un método totalmente genérico, que se puede aplicar a una amplia gama de topologías de filtros diferentes.

En el siguiente capítulo se explicará la aplicación de dicho método para la síntesis dimensional de un tipo concreto de filtros, los filtros de modo evanescente. También se detallará la implementación realizada, que permite a un usuario realizar la síntesis de una forma totalmente automatizada.

En el cuarto capítulo se mostrarán una serie de resultados obtenidos con la aplicación desarrollada, mostrando las grandes ventajas del método propuesto.

El último capítulo de esta memoria incluye las principales conclusiones que se pueden extraer del trabajo desarrollado, así como las líneas de trabajo futuras que éste abre.

Capítulo 2

Método del slope parameter

2.1. Prototipo paso bajo en línea

Consideremos inicialmente el prototipo paso bajo equivalente de elementos concentrados de orden n, realizado con bobinas en serie y condensadores en paralelo. Los elementos de dicho prototipo están caracterizados por las parámetros $g_{i_{\mathrm{N}}}$, obtenidos con las técnicas clásicas y según el tipo de respuesta que se desee sintetizar (Butterworth, Chebychev, ...). El subíndice N hace referencia a que se tratan de valores normalizados a una impedancia/admitancia de fuente de valor unidad. En la Figura 2.1 se muestra la configuración que comienza con una bobina serie, de modo que $g_{0_{\mathrm{N}}}$ representa la admitancia de fuente, que al estar normalizada será siempre igual a 1. Por su parte, $g_{n+1_{\mathrm{N}}}$ denota a la impedancia (para n par) o admitancia (para n impar) de carga normalizada, y cuyo valor puede ser en general diferente a 1.

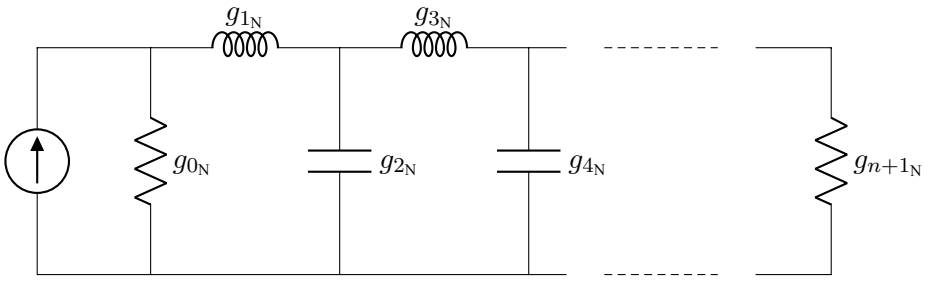
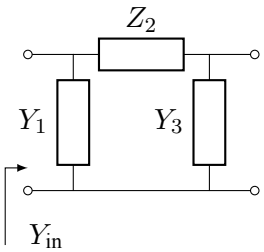


Figura 2.1: Prototipo paso bajo de orden n con coeficientes g_i normalizados.

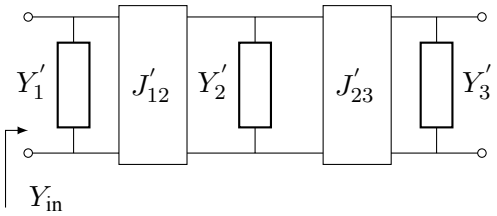
La finalidad es ahora llegar a un esquemático en el que únicamente tengamos elementos en paralelo. Teniendo en cuenta que la respuesta a nivel de parámetros S del circuito viene fijada por la admitancia de fuente (que se siempre la misma) y la admitancia de entrada, estableceremos la equivalencia a nivel de admitancias de entrada mostrada en la Figura 2.2.

La admitancia de entrada en la red asociada a la Figura 2.2a es:

$$Y_{\mathrm{in}} = Y_1 + \frac{1}{Z_2 + \dfrac{1}{Y_3}}$$



(a) Red en Π



(b) Equivalente con inversores de admitancia

Figura 2.2: Equivalencia entre red en Π y red con admitancias en paralelo interconectadas con inversores inmitancias.

y la de la Figura 2.2b viene dada por:

$$Y_{\text{in}} = Y_1' + \frac{J_{12}'^2}{Y_2' + \frac{J_{23}'^2}{Y_3'}} = Y_1' + \frac{1}{\frac{Y_2'}{J_{12}'^2} + \frac{J_{23}'^2}{J_{12}'^2 Y_3'}}$$

Tras identificar términos en ambas expresiones para establecer la equivalencia, podemos deducir que:

$$\begin{aligned} Y_1' &= Y_1 \\ Y_2' &= J_{12}'^2 Z_2 \\ Y_3' &= \frac{J_{23}'^2 Y_3}{J_{12}'^2} \end{aligned}$$

De esta forma, en el prototipo inicial que teníamos en la Figura 2.1, los elementos resonantes en serie pasaran a ser dos inversores de admitancia y una condensador paralelo, obteniendo el prototipo mostrado en la Figura 2.3. de forma que los primeros coeficientes g_i' vendrán dados por:

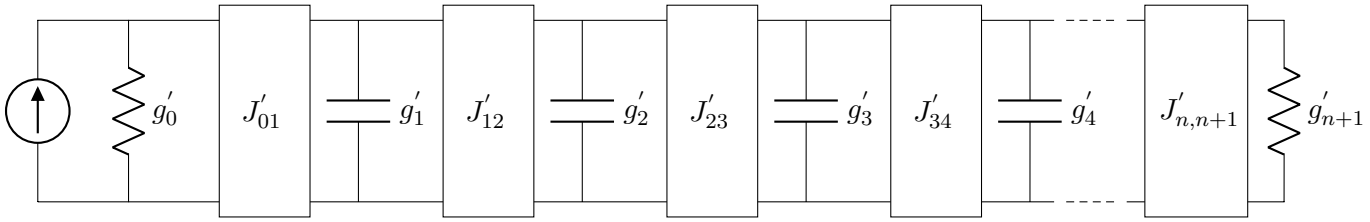


Figura 2.3: Prototipo paso bajo transformado a paralelo.

$$
\begin{aligned}
g_0^{'} &= g_{0_N} = 1 & g_1^{'} &= J_{01}^{'2} g_{1_N} \\
g_2^{'} &= \frac{J_{12}^{'2}}{J_{01}^{'2}} g_{2_N} & g_3^{'} &= \frac{J_{23}^{'2} J_{01}^{'2}}{J_{12}^{'2}} g_{3_N} \\
g_4^{'} &= \frac{J_{34}^{'2} J_{12}^{'2}}{J_{23}^{'2} J_{01}^{'2}} g_{4_N} & g_5^{'} &= ...
\end{aligned}
$$

lo que nos permite deducir la expresión general (2.1) para dichos coeficientes:

$$
\begin{aligned}
g_{2m-1}^{'} &= \frac{\prod_{p=0}^{m-1} J_{2p,2p+1}^{'2}}{\prod_{p=1}^{m-1} J_{2p-1,2p}^{'2}} g_{2m-1_N}, \text{ con } m = \begin{cases} 1,2,3,\cdots,\dfrac{n}{2}+1 \text{ si } n \text{ es par} \\ 1,2,3,\cdots,\dfrac{n+1}{2} \text{ si } n \text{ es impar} \end{cases} \\
g_{2m}^{'} &= \frac{\prod_{p=1}^{m} J_{2p-1,2p}^{'2}}{\prod_{p=0}^{m-1} J_{2p,2p+1}^{'2}} g_{2m_N}, \text{ con } m = \begin{cases} 0,1,2,3,\cdots,\dfrac{n}{2} \text{ si } n \text{ es par} \\ 0,1,2,3,\cdots,\dfrac{n+1}{2} \text{ si } n \text{ es impar} \end{cases}
\end{aligned}
\tag{2.1}
$$

y cuyo valor depende de las constantes de inversión $J_{i,i+1}$ que pueden ser libremente elegidas.

Las expresiones de los coeficientes obtenidos asumen que la admitancia de fuente $g_{0_N} = 1$. No obstante, resulta mejor generalizar los resultados para el caso de un valor arbitrario g_0 de la admitancia de fuente. Este hecho supone multiplicar las admitancias por un factor g_0 y las impedancias por un factor $1/g_0$.

Mediante este ultimo paso, conseguiremos la forma general del prototipo en línea paso bajo equivalente mostrada en la Figura 2.4 y donde las expresiones de los coeficientes g_i son, también en general:

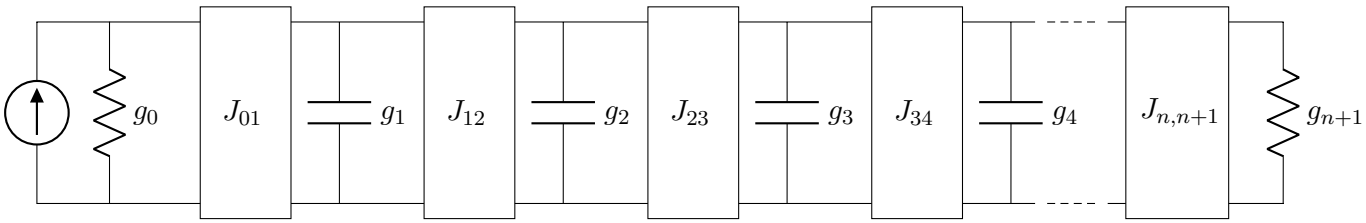


Figura 2.4: Prototipo paso bajo paralelo equivalente generalizado de orden n.

$$
\begin{aligned}
g_{2m-1}^{'} &= \frac{\prod_{p=0}^{m-1} J_{2p,2p+1}^{'2}}{\prod_{p=1}^{m-1} J_{2p-1,2p}^{'2}} \frac{g_{2m-1_N}}{g_0}, \text{ con } m = \begin{cases} 1,2,3,\cdots,\dfrac{n}{2}+1 \text{ si } n \text{ es par} \\ 1,2,3,\cdots,\dfrac{n+1}{2} \text{ si } n \text{ es impar} \end{cases} \\
g_{2m}^{'} &= \frac{\prod_{p=1}^{m} J_{2p-1,2p}^{'2}}{\prod_{p=0}^{m-1} J_{2p,2p+1}^{'2}} g_{2m_N} g_0, \text{ con } m = \begin{cases} 0,1,2,3,\cdots,\dfrac{n}{2} \text{ si } n \text{ es par} \\ 0,1,2,3,\cdots,\dfrac{n+1}{2} \text{ si } n \text{ es impar} \end{cases}
\end{aligned}
\tag{2.2}
$$

Mediante las expresiones en (2.2) podemos ver como los inversores de admitancia nos introducirán grados de libertad adicionales en el prototipo. No obstante, en vez de elegir el valor de $J_{i,i+1}$, es mucho más práctico que el diseñador fije el valor de los parámetros g_i del prototipo para que se

parezcan a los elementos reales del circuito a los que representan. Por lo tanto, para recuperar la respuesta deseada, las constantes de inversión de los inversores de admitancia se deberán obtener como:

$$g_1 = g_{2m-1|_{m=1}} = J_{01}^2 \frac{g_{1_N}}{g_0} \longrightarrow J_{01} = \sqrt{\frac{g_0 g_1}{g_{1_N}}}$$
$$g_2 = g_{2m|_{m=1}} = \frac{J_{12}^2}{J_{01}^2} g_{2_N} g_0 = J_{12}^2 \frac{g_{1_N} g_{2_N}}{g_1} \longrightarrow J_{12} = \sqrt{\frac{g_1 g_2}{g_{1_N} g_{2_N}}}$$
$$g_3 = g_{2m-1|_{m=2}} = \frac{J_{01}^2 J_{23}^2}{J_{12}^2} \frac{g_{3_N}}{g_0} = J_{23}^2 \frac{g_{2_N} g_{3_N}}{g_2} \longrightarrow J_{23} = \sqrt{\frac{g_2 g_3}{g_{2_N} g_{3_N}}}$$
$$g_4 = g_{2m|_{m=2}} = \frac{J_{12}^2 J_{34}^2}{J_{01}^2 J_{23}^2} g_{4_N} g_0 = J_{34}^2 \frac{g_{3_N} g_{4_N}}{g_3} \longrightarrow J_{34} = \sqrt{\frac{g_3 g_4}{g_{3_N} g_{4_N}}}$$

y actuando por inducción, llegamos a la expresión general para las constantes de los inversores de admitancias:

$$J_{i,i+1} = \sqrt{\frac{g_i g_{i+1}}{g_{i_N} g_{i+1_N}}} \qquad \text{, con } i = 0, 1, 2, \cdots, n \tag{2.3}$$

Esta flexibilidad a la hora del diseño, derivada de la introducción de los inversores de inmitancias, resulta muy interesante en la práctica ya que permite fijar el valor de los elementos concentrados en paralelo para obtener la mayor similitud con la estructura real. Esta es precisamente la principal contribución del método propuesto por Cohn en [4].

2.2. Prototipo paso banda en línea

La transformación en frecuencia que normalmente se emplea para pasar de una respuesta paso bajo equivalente normalizada (es decir, con pulsación de corte $\omega_c^{'} = 1$) a una respuesta paso banda con pulsaciones de corte de la banda de paso $\omega_1 = 2\pi f_1$ y $\omega_2 = 2\pi f_2$ es:

$$\omega^{'} \longrightarrow \frac{1}{\Delta}\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right)$$

donde tanto la pulsación central ω_0 (correspondiente a un valor de $\omega^{'} = 0$ en la respuesta paso bajo equivalente) como el parámetro Δ son incógnitas a determinar.

Teniendo presente que las pulsaciones de corte de la banda de paso del filtro se deben corresponder con la pulsación de corte de la respuesta paso bajo normalizada ($\omega_c^{'} = \pm 1$, al ser simétrica respecto al origen de eje en frecuencias), se obtienen las dos igualdades que necesitaremos para obtener las dos incógnitas del mapeado en frecuencia:

$$\left.\begin{aligned} \frac{1}{\Delta}\left(\frac{\omega_1}{\omega_0} - \frac{\omega_0}{\omega_1}\right) &= -1 \\ \frac{1}{\Delta}\left(\frac{\omega_2}{\omega_0} - \frac{\omega_0}{\omega_2}\right) &= +1 \end{aligned}\right\}$$ y si dividimos los términos entre ellos y despejamos nos queda:

$$\left(\frac{\omega_1}{\omega_0}-\frac{\omega_0}{\omega_1}\right)=-\left(\frac{\omega_2}{\omega_0}-\frac{\omega_0}{\omega_2}\right)\longrightarrow\frac{\omega_1^2-\omega_0^2}{\omega_1\omega_0}=\frac{\omega_0^2-\omega_2^2}{\omega_2\omega_0}$$
$$\omega_2\omega_1^2-\omega_2\omega_0^2=\omega_1\omega_0^2-\omega_1\omega_2^2$$
$$\omega_0^2(\omega_1+\omega_2)=\omega_2\omega_1^2+\omega_2^2\omega_1=\omega_2\omega_1(\omega_1+\omega_2)$$
$$\omega_0=\sqrt{\omega_1\omega_2}$$

lo que nos ha permitido obtener la expresión de la pulsación central del filtro $\omega_0=2\pi f_0$. Así mismo, a partir del conocimiento de este parámetro, ya podemos obtener el valor de Δ con cualquiera de las dos expresiones anteriores. Por ejemplo, empleando la segunda:

$$\frac{1}{\Delta}\left(\frac{\omega_2}{\omega_0}-\frac{\omega_0}{\omega_2}\right)=1\longrightarrow\Delta=\frac{\omega_2}{\omega_0}-\frac{\omega_0}{\omega_2}=\sqrt{\frac{\omega_2}{\omega_1}}-\sqrt{\frac{\omega_1}{\omega_2}}=\frac{\omega_2-\omega_1}{\sqrt{\omega_1\omega_2}}$$
$$\Delta=\frac{\omega_2-\omega_1}{\omega_0}$$

se obtiene que el parámetro Δ representa el ancho de banda relativo de la banda de paso.

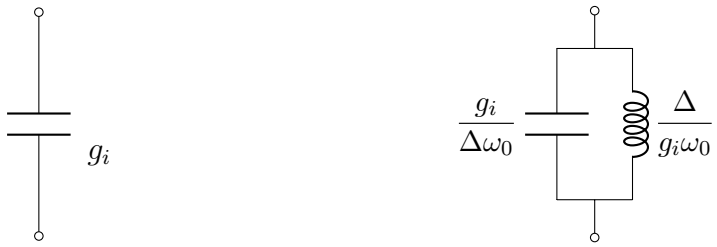
Una vez determinados los parámetros incógnita, ya podemos construir por completo el mapeado en frecuencia que usaremos para pasar de la respuesta paso bajo equivalente normalizada (implementada con el prototipo detallado en la Sección 2.1) a la respuesta paso banda deseada (con banda de paso comprendida entre f_1 y f_2):

$$\omega'\longrightarrow\frac{1}{\Delta}\left(\frac{\omega}{\omega_0}-\frac{\omega_0}{\omega}\right)$$
$$\omega_0=2\pi f_0=2\pi\sqrt{f_1f_2}$$
$$\Delta=\frac{f_2-f_1}{f_0}\tag{2.4}$$

Al aplicar el mapeado en frecuencia, los elementos paralelo del prototipo paso bajo experimentaran a su vez una transformación. De esta forma, la admitancia del i-ésimo elemento pasará a ser:

$$Y_i=\mathrm{j}\omega' g_i=\mathrm{j}\frac{1}{\Delta}\left(\frac{\omega}{\omega_0}-\frac{\omega_0}{\omega}\right)g_i=\mathrm{j}\omega\frac{g_i}{\Delta\omega_0}+\frac{1}{\mathrm{j}\omega\frac{\Delta}{g_i\omega_0}}=\mathrm{j}\omega C_i+\frac{1}{\mathrm{j}\omega L_i}=\mathrm{j}B_i\tag{2.5}$$

es decir, una bobina y un condensador en paralelo, y que gráficamente se representa en la Figura 2.5.



(a) Prototipo paso bajo **(b) Prototipo paso banda**

Figura 2.5: Transformación de los elementos paralelo del filtro paso bajo al aplicarles el mapeado en frecuencia.

Por lo tanto, al hacer la transformación paso bajo a paso banda, cada condensador en paralelo se transforma en un circuito resonante compuesto por un condensador y una bobina en paralelo. La pulsación de resonancia de cada uno de estos elementos será igual a la central del filtro ω_0 ((2.4)) independientemente del valor que se tenga en g_i:

$$\frac{1}{\sqrt{L_i C_i}} = \frac{1}{\sqrt{\frac{\Delta}{g_i \omega_0}\frac{g_i}{\Delta\omega_0}}} = \sqrt{\omega_0^2} = \omega_0 \tag{2.6}$$

A primera vista, se puede presuponer que los parámetros g_i del prototipo paso bajo equivalente no tienen importancia en el diseño de un filtro paso banda. No obstante, hay que tener en cuenta que estos parámetros son los que precisamente están asociados al tipo de respuesta del dispositivo, y por tanto deben ser relevantes de alguna forma.

Los parámetros g_i aparecen en el valor del condensador y la bobina del elemento resonante transformado, y aunque no influyen sobre la frecuencia de resonancia de este elemento, si que deben afectar a la admitancia que presentan según la expresión (2.5). A la frecuencia de resonancia la admitancia del elemento se anula por definición, pero la forma en la que varía la admitancia en función de la frecuencia si que debe estar afectado por g_i. De hecho, para el i-ésimo resonador obtenido tras la transformación en frecuencia, obtenemos:

$$\frac{dB_i}{d\omega} = \frac{g_i}{\Delta}\left(\frac{1}{\omega_0} + \frac{\omega_0}{\omega^2}\right) \tag{2.7}$$

El parámetro de pendiente o *slope parameter* de un resonador justamente mide esa variación entorno a la frecuencia central del resonador, al estar definido mediante:

$$b \triangleq \frac{\omega_0}{2}\left.\frac{dB}{d\omega}\right|_{\omega_0} \tag{2.8}$$

y aplicándolo al i-ésimo resonador de la estructura, obtenemos:

$$b_i = \frac{\omega_0}{2}\left.\frac{dB_i}{d\omega}\right|_{\omega_0} = \frac{\omega_0}{2\Delta} g_i \left.\left(\frac{1}{\omega_0} + \frac{\omega_0}{\omega^2}\right)\right|_{\omega_0} = \frac{\omega_0}{2\Delta} g_i \left(\frac{2}{\omega_0}\right) = \frac{g_i}{\Delta} = \sqrt{\frac{C_i}{L_i}}$$

es decir

$$g_i = \Delta b_i \tag{2.9}$$

La ecuación (2.9) refleja que el valor de g_i va ligado al slope parameter b_i del resonador real de la estructura que lo implemente (excepto los parámetros g_i de entrada y salida, es decir, g_0 y g_{n+1}, que van asociados a las admitancias de los puertos).

En este momento ya disponemos de todos los datos para construir el prototipo en línea paso banda de elementos concentrados. Partiendo del prototipo paso bajo equivalente normalizado de la Figura 2.4 y aplicando la transformación en frecuencia (2.4), obtenemos el circuito equivalente mostrado en la Figura 2.6 cuyos diferentes parámetros son constantes con la frecuencia, y su valor viene dado por:

$$\omega_0 = 2\pi f_0 = 2\pi\sqrt{f_1 f_2} \quad , \quad \Delta = \frac{f_2 - f_1}{f_0} \tag{2.10}$$

$$g_0 = Y_{0_{\text{in}}}|_{\omega_0} \quad , \quad g_i = \Delta b_i (i = 1, 2, \cdots, n) \quad , \quad g_{n+1} = Y_{0_{\text{out}}}|_{\omega_0} \tag{2.11}$$

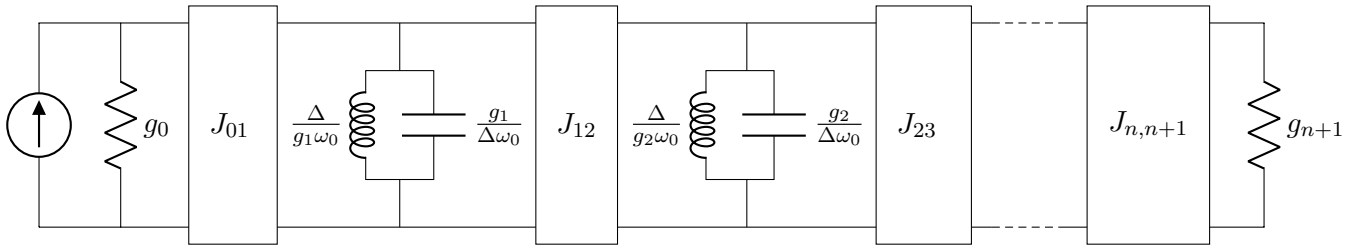


Figura 2.6: Filtro transformado a paso banda.

donde b_i denota al *slope parameter* del i-ésimo resonador de la estructura, mientras que $Y_{0_{\text{in}}}$ e $Y_{0_{\text{out}}}$ son las admitancias de los puertos de entrada y de salida. Finalmente, atendiendo a (2.3) y (2.11), las constantes de inversión de los inversores de admitancia son:

$$J_{01} = \sqrt{\frac{Y_{0_{\text{in}}}|_{\omega_0}\,\Delta b_1}{g_{0_{\text{N}}} g_{1_{\text{N}}}}} \quad , \quad J_{i,i+1} = \Delta\sqrt{\frac{b_i b_{i+1}}{g_{i_{\text{N}}} g_{i+1_{\text{N}}}}} \quad , \quad J_{n,n+1} = \sqrt{\frac{\Delta b_n\; Y_{0_{\text{out}}}|_{\omega_0}}{g_{n_{\text{N}}} g_{n+1_{\text{N}}}}} \tag{2.12}$$

El procedimiento se ha desarrollado para obtener un prototipo con resonadores en paralelo acoplados mediante inversores de admitancia. Se podría aplicar una técnica dual para obtener una representación equivalente con resonadores en serie acoplados mediante inversores de impedancia, obteniendo en ese caso las expresiones duales. De esta forma, las constantes de inversión de los inversores de impedancia serán:

$$K_{01} = \sqrt{\frac{Z_{0_{\text{in}}}|_{\omega_0}\,\Delta x_1}{g_{0_{\text{N}}} g_{1_{\text{N}}}}} \quad , \quad K_{i,i+1} = \Delta\sqrt{\frac{x_i x_{i+1}}{g_{i_{\text{N}}} g_{i+1_{\text{N}}}}} \quad , \quad K_{n,n+1} = \sqrt{\frac{\Delta x_n\; Z_{0_{\text{out}}}|_{\omega_0}}{g_{n_{\text{N}}} g_{n+1_{\text{N}}}}} \tag{2.13}$$

siendo $Z_{0_{\text{in}}}$ e $Z_{0_{\text{out}}}$ las impedancias de los puertos de entrada y de salida de la estructura real, y x_i el parámetro de pendiente del i-esimo resonador asociado a su reactancia, definido por:

$$x_i = \frac{\omega_0}{2}\,\frac{dX_i}{d\omega}\bigg|_{\omega_0} \tag{2.14}$$

2.3. Resonador cargado

Consideremos ahora el caso de un resonador de una estructura genérica (con frecuencia de resonancia f_0), conectada a sus puertos de entrada y de salida mediante sus respectivos elementos de acoplo (ver Figura 2.7).

Vamos a suponer que conocemos los parámetros S de dicha estructura, bien por haberlos medido (para el caso de una estructura real fabricada) o bien por haberlos obtenido por simulación (si se trata de un modelo electromagnético de simulación). Lo que tenemos que hacer es intentar representar dicha estructura a través de un modelo circuital con la misma respuesta y que sea similar al empleado para representar un filtro (ver la Figura 2.6). En concreto, si nos centramos en un resonador del prototipo, el equivalente circuital que vamos a considerar es el mostrado en la

Figura 2.7: Estructura genérica de microondas con un resonador.

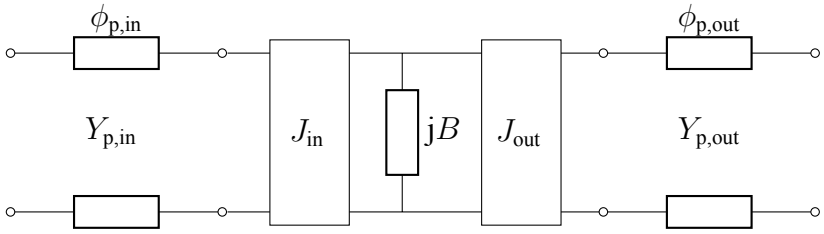


Figura 2.8: Prototipo circuital equivalente de la estructura considerada.

Figura 2.8, donde se han añadido unos tramos de línea de entrada y salida para poder representar los desfases en los puertos de entrada y salida de la estructura.

Calculemos los parámetros S del equivalente circuital sin tener en cuenta las líneas de transmisión de entrada y salida (ver la Figura 2.9).

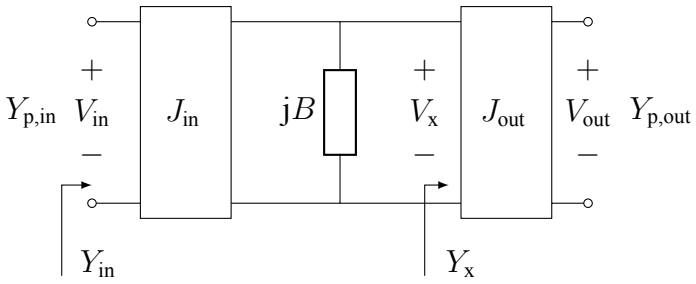


Figura 2.9: Prototipo circuital equivalente sin desfases.

El parámetro S_{11} se puede obtener fácilmente a partir de la admitancia de entrada Y_{in} asumiendo que el puerto de salida está terminado (es decir, cargado con una admitancia $Y_{p,out}$):

$$Y_{in} = \frac{J_{in}^2}{jB + Y_x} = \frac{J_{in}^2}{jB + \dfrac{J_{out}^2}{Y_{p,out}}} = \frac{J_{in}^2 Y_{p,out}}{jBY_{p,out} + J_{out}^2}$$

de donde se deduce que:

$$S_{11} = \frac{Y_{\text{p,in}} - Y_{\text{in}}}{Y_{\text{p,in}} + Y_{\text{in}}} = \frac{Y_{\text{p,in}} - \dfrac{J_{\text{in}}^2 Y_{\text{p,out}}}{\text{j}BY_{\text{p,out}} + J_{\text{out}}^2}}{Y_{\text{p,in}} + \dfrac{J_{\text{in}}^2 Y_{\text{p,out}}}{\text{j}BY_{\text{p,out}} + J_{\text{out}}^2}} = \frac{Y_{\text{p,in}}\left(\text{j}BY_{\text{p,out}} + J_{\text{out}}^2\right) - J_{\text{in}}^2 Y_{\text{p,out}}}{Y_{\text{p,in}}\left(\text{j}BY_{\text{p,out}} + J_{\text{out}}^2\right) + J_{\text{in}}^2 Y_{\text{p,out}}} =$$
$$= \frac{J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}} \tag{2.15}$$

Así mismo, podremos deducir el valor del coeficiente de transmisión S_{21} mediante la expresión clásica que lo relaciona con S_{11}:

$$S_{21} = \frac{V_{\text{out}}}{V_{\text{in}}}\sqrt{\frac{Y_{\text{p,out}}}{Y_{\text{p,in}}}}\,(1 + S_{11}) = \frac{V_{\text{out}}}{V_{\text{in}}}\sqrt{\frac{Y_{\text{p,out}}}{Y_{\text{p,in}}}}\,\frac{2\left(J_{\text{out}}^2 Y_{\text{p,in}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}\right)}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}}$$

lo que require que determinemos el cociente entre V_{in} y V_{out}. Puesto que estamos trabajando con inversores, podemos extraerlo haciendo uso de la matriz Y de un inversor de admitancias:

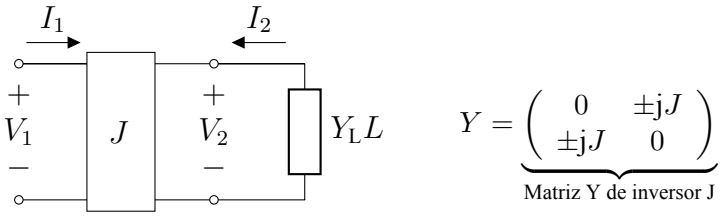


Figura 2.10: Inversor J y su matriz de admitancias.

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 0 & \pm\text{j}J \\ \pm\text{j}J & 0 \end{bmatrix}\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \longrightarrow \begin{array}{l} I_1 = \pm\text{j}JV_2 \\ I_2 = \pm\text{j}JV_1 \end{array}$$

del circuito correspondiente al inversor J obtenemos que $I_2 = -Y_{\text{L}}V_2$, y por tanto podemos plantear que:

$$-Y_{\text{L}}V_2 = \pm\text{j}JV_1 \longrightarrow \frac{V_1}{V_2} = \frac{-Y_{\text{L}}L}{\pm\text{j}J}$$

y aplicando esta relación al circuito de la Figura 2.9, es posible extraer la relación entre su tensión de entrada y de salida:

$$\left.\begin{aligned} V_{\text{x}} &= -\frac{Y_{\text{p,out}}}{\pm\text{j}J_{\text{out}}}V_{\text{out}} \\ V_{\text{in}} &= -\frac{\left(\text{j}B + \dfrac{J_{\text{out}}^2}{Y_{\text{p,out}}}\right)}{\pm\text{j}J_{\text{in}}}V_{\text{x}} \end{aligned}\right\} \quad V_{\text{in}} = \frac{\text{j}B + \dfrac{J_{\text{out}}^2}{Y_{\text{p,out}}}}{\pm jJ_{\text{in}}}\frac{Y_{\text{p,out}}}{\pm\text{j}J_{\text{out}}}V_{\text{out}} = -\frac{\text{j}BY_{\text{p,out}} + J_{\text{out}}^2}{J_{\text{in}}J_{\text{out}}}V_{\text{out}}$$

donde se ha asumido que los dos inversores de admitancias tienen el mismo signo. Seguidamente, podremos obtener la expresión del parámetro S_{21}:

$$S_{21} = -\frac{J_{\text{in}}J_{\text{out}}}{\text{j}BY_{\text{p,out}} + J_{\text{out}}^2)}\sqrt{\frac{Y_{\text{p,out}}}{Y_{\text{p,in}}}}\frac{2(J_{\text{out}}^2 Y_{\text{p,in}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}})}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}} =$$
$$= -\frac{2\sqrt{Y_{\text{p,in}}Y_{\text{p,out}}}J_{\text{in}}J_{\text{out}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}} \tag{2.16}$$

Finalmente, para obtener las expresiones finales de los parámetros S del circuito equivalente de la Figura 2.8 únicamente quedan por añadir los desfases que introducen las lineas de transmisión de entrada y salida. Cabe recalcar que el parámetro S_{12} se obtiene con la propiedad de reciprocidad a partir del S_{21}, y el S_{22} intercambiando en la expresión del S_{11} los parámetros del puerto 1 y del puerto 2 entre sí:

$$S_{11} = e^{-2\text{j}\phi_{\text{in}}}\frac{J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}}$$
$$S_{22} = e^{-2\text{j}\phi_{\text{out}}}\frac{J_{\text{in}}^2 Y_{\text{p,out}} - J_{\text{out}}^2 Y_{\text{p,in}} + \text{j}BY_{\text{p,in}}Y_{\text{p,out}}}{J_{\text{in}}^2 Y_{\text{p,out}} + J_{\text{out}}^2 Y_{\text{p,in}} + \text{j}BY_{\text{p,in}}Y_{\text{p,out}}}$$
$$S_{21} = S_{12} = -e^{-j(\phi_{\text{in}}+\phi_{\text{out}})}\frac{2\sqrt{Y_{\text{p,in}}Y_{\text{p,out}}}J_{\text{in}}J_{\text{out}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}BY_{\text{p,out}}Y_{\text{p,in}}} \tag{2.17}$$

Como el circuito equivalente de la Figura 2.8 es recíproco, pasivo y sin pérdidas, cumple por tanto la condición de unitariedad. Como resultado, de los 6 grados de libertad de la expresión (2.17) (el módulo y la fase de cada parámetro S), sólo quedan realmente 3 libres (ya que $|S_{11}| = |S_{22}|$, $|S_{21}|^2 = 1 - |S_{11}|^2$, y la fase del parámetro S_{22} viene fijada a partir de la fase de los parámetros S_{11} y S_{21} mediante $\varphi_{22} = 2\varphi_{21} - \varphi_{11} \pm \pi$). Por lo tanto, de todos los grados de libertad de los que dispone el equivalente circuital, es decir, J_{in}, J_{out}, B, $Y_{\text{p,in}}$, $Y_{\text{p,out}}$, $\phi_{\text{p,in}}$ y $\phi_{\text{p,out}}$, sólo necesitamos un total de 3 para ajustar los parámetros S de la estructura.

Los valores de J_{in} y J_{out} se pueden fijar a partir de los parámetros S del elemento real de acoplo de entrada y salida del resonador considerado de forma aislada, respectivamente, aplicando la conocida relación válida para elementos pasivos, recíprocos y sin pérdidas:

$$|S_{11}| = \frac{1-\bar{J}^2}{1-\bar{J}^2} \quad \longrightarrow \quad \bar{J} = \frac{J}{\sqrt{Y_{01}Y_{02}}} = \frac{1}{\sqrt{Y_{01}Y_{02}}}\sqrt{\frac{1-|S_{11}|}{1+|S_{11}|}} \tag{2.18}$$

donde Y_{01} e Y_{02} son la admitancia del puerto de entrada y de salida del elemento de acoplo considerado de forma aislada. El ajuste se realizará a la frecuencia de resonancia de la cavidad f_0 (obteniendo un valor constante para J_{in} y J_{out}), ya que el prototipo circuital equivalente pasobanda de la Figura 2.6 está formado por inversores de admitancia constantes con la frecuencia.

Así mismo, las admitancias de los puertos de entrada y de salida se fijarán igual a la de los puertos de la estructura real a la frecuencia f_0 de resonancia de la estructura real, es decir:

$$Y_{\text{p,in}} = Y_{\text{e,in}}|_{f=f_0} \quad , \quad Y_{\text{p,out}} = Y_{\text{e,out}}|_{f=f_0} \tag{2.19}$$

logrando así un equivalencia con la estructura real a nivel de admitancias de los puertos e inversores de inmitancias exacta a la frecuencia de resonancia f_0 del resonador. Así mismo, es la mejor

forma de actuar para lograr una equivalencia con el prototipo paso banda cuyos elementos son independientes de la frecuencia.

Una vez fijado estos parámetros, nos quedan B, $\phi_{\text{p,in}}$ y $\phi_{\text{p,out}}$ para poder ajustar los 3 grados de libertad de los parámetros S de la estructura. Teniendo en cuenta que los parámetros S varían con la frecuencia, estos 3 parámetros del prototipo también tendrán que variar con la frecuencia para mantener la misma respuesta. Es evidente que $\phi_{\text{p,in}}$ y $\phi_{\text{p,out}}$ serán los encargados de ajustar la fase de dos de los parámetros S, por ejemplo S_{11} y S_{21}. Por lo tanto, la función del parámetro B es realizar el ajuste del módulo de los parámetros S de la estructura real.

La forma más simple de obtener el parámetro B es mediante $|S_{11}|$. A partir de (2.17) se tiene que

$$|S_{11}|^2 = \frac{(J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}})^2 + B^2 Y_{\text{p,in}}^2 Y_{\text{p,out}}^2}{(J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}})^2 + B^2 Y_{\text{p,in}}^2 Y_{\text{p,out}}^2}$$

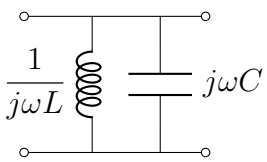
despejando ahora B^2:

$$B^2(f) = \frac{(J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}})^2 - |S_{11}(f)|^2 (J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}})^2}{Y_{\text{p,in}}^2 Y_{\text{p,out}}^2 (|S_{11}(f)|^2 - 1)}$$

y aplicando la raiz cuadrada, se obtiene el valor de la susceptancia $B\,(f)$

$$B(f) = \frac{\text{sgn}\,(f - f_0)}{Y_{\text{p,in}} Y_{\text{p,out}}} \sqrt{\frac{|S_{11}(f)|^2 (J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}})^2 - (J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}})^2}{1 - |S_{11}(f)|^2}} \tag{2.20}$$

donde se ha empleado la función signo para tener en cuenta que para frecuencias por debajo de su frecuencia de resonancia f_0 la susceptancia tiene un caracter inductivo, y para frecuencias superiores su carácter es capacitivo, tal y como corresponde con un circuito resonante paralelo (ver la Figura 2.11)



- Para $f = f_0, \quad B = 0$
- Para$f > f_0$ domina j$\omega C, \quad B > 0$
- Para$f < f_0$ domina $\frac{1}{\text{j}\omega L}, \quad B < 0$

Figura 2.11: Comportamiento de circuito resonante en paralelo.

Una vez que se ha obtenido el valor de la susceptancia $B(f)$ que hace que el prototipo circuital equivalente tenga el mismo módulo de los parámetros S que la estructura real, ya podemos ajustar el valor de los desfases $\phi_{\text{p,in}}$ y $\phi_{\text{p,out}}$ para ajustar también la respuesta en fase. A partir nuevamente de Ecuación 2.17 obtenemos:

$$\phi_{\text{p,in}}(f) = \frac{1}{2}\angle\left\{\frac{1}{S_{11}(f)}\frac{J_{\text{out}}^2 Y_{\text{p,in}} - J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}B(f)Y_{\text{p,in}}Y_{\text{p,out}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}B(f)Y_{\text{p,in}}Y_{\text{p,out}}}\right\}$$
$$\phi_{\text{p,out}}(f) = \frac{1}{2}\angle\left\{\frac{1}{S_{21}(f)}\frac{2\sqrt{Y_{\text{p,in}}Y_{\text{p,out}}}J_{\text{in}}J_{\text{out}}}{J_{\text{out}}^2 Y_{\text{p,in}} + J_{\text{in}}^2 Y_{\text{p,out}} + \text{j}B(f)Y_{\text{p,in}}Y_{\text{p,out}}}\right\} - \phi_{\text{p,in}}(f) \pm \pi \qquad (2.21)$$

Las expresiones recién obtenidas permiten obtener los parámetros $B(f)$, $\phi_{\text{p,in}}(f)$ y $\phi_{\text{p,out}}(f)$ del circuito equivalente mediante los parámetros S obtenidos tras excitar sólo por el puerto 1 de la estructura (puerto de entrada). En el caso de que sólo sea posible o se quiera excitar la estructura por el puerto de salida, se pueden usar las mismas expresiones intercambiando $J_{\text{in}} \leftrightharpoons J_{\text{out}}$, $Y_{\text{p,in}} \leftrightharpoons Y_{\text{p,out}}$, $S_{11} \leftrightharpoons S_{22}$, $S_{21} \leftrightharpoons S_{12}$ y $\phi_{\text{p,in}} \leftrightharpoons \phi_{\text{p,out}}$.

Como resultado de todo lo anterior, ya se puede construir el circuito equivalente de una estructura real formada por un resonador cargado con sus elementos de acoplo de entrada y de salida. Además, dicho circuito equivalente se puede trasladar al prototipo de la Figura 2.6 obtenido en la Sección 2.2, ya que los inversores de admitancias son constantes con la frecuencia. Obsérvese también como un mismo elemento de acoplo está asociado a dos resonadores en la estructura real, pero al estar todos los resonadores sintonizados a la misma frecuencia f_0, el valor de la constante de inversión obtenida será el mismo en ambos casos, siendo de esta forma consistente y compatible con el prototipo del filtro paso banda. Así mismo, los desfases de entrada y salida (al ser parte de los resonadores adyacentes o de los puertos de la estructura) no son relevantes de cara al prototipo pasobanda equivalente. No obstante, el parámetro que es crucial aquí es la susceptancia $B\,(f)$, al representar la variación con la frecuencia del resonador cargado con sus elementos de acoplo (al considerar los puertos y los inversores de admitancias constantes con la frecuencia - como sucede con el prototipo -, la variación con la frecuencia de dichos elementos reales se han trasladado precisamente a esta susceptancia). En el prototipo pasobanda equivalente dicha variación se introduce a partir del *slope parameter* del resonador, que se podrá calcular como:

$$b = \frac{\omega_0}{2}\left.\frac{dB}{d\omega}\right|_{\omega_0} = \frac{f_0}{2}\left.\frac{dB}{df}\right|_{f_0} \simeq \frac{f_0}{2}\frac{B(f_0 - h) - B(f_0 + h)}{2h} \qquad (2.22)$$

y que, a diferencia de los métodos clásicos, ahora sí que tiene en cuenta el comportamiento real del resonador así como el efecto de carga originado por los elementos de acoplo con los que se conecta al resto del circuito. De esta forma, el grado de correspondencia entre la estructura real y el prototipo circuital equivalente es muy alta.

Con lo descrito en esta última sección se puede pasar de un filtro real a un prototipo paso banda, al poder establecer la equivalencia resonador a resonador entre la estructura real y su prototipo circuital equivalente. Sin embargo, ahora quedaría realizar el paso contrario, es decir, extraer una estructura real que implemente un prototipo pasobanda sintetizado previamente para proporcionar la respuesta eléctrica deseada. El disponer de una muy buena correspondencia entre la estructura real y el prototipo circuital debería permitirnos la obtención de unas mejores dimensiones iniciales de la estructura, y por tanto reducir el esfuerzo de optimización final. Este es precisamente el objeto del próximo capítulo de esta memoria.

Capítulo 3

Aplicación e implementación

3.1. Introducción

El procedimiento teórico descrito en el Capítulo 2 proporciona una novedosa vía de obtención del *slope parameter* que además de mejorar la precisión, en comparación a [8], permite tener en cuenta efectos de orden superior, así como la dependencia en frecuencia conjunta de los resonadores reales y sus elementos de acoplo, integrantes de las estructuras de microondas. Ello permite establecer una semejanza mucho más cercana a los prototipos circuitales y disminuir el coste de la optimización, tercer paso descrito en el Subsección 1.1.3, o en un mejor caso, evitar incluso dicha optimización.

3.1.1. *Slope parameter* clásico

Tradicionalmente el valor del *slope parameter* de cada resonador es obtenido teóricamente en base a su frecuencia de resonancia f_0:

$$b_i = \frac{\omega}{2} \left. \frac{dB_i}{d\omega} \right|_{\omega=\omega_0} \tag{3.1}$$

En el proceso clásico se consideran los resonadores como aislados e ideales, de modo que una cavidad rectangular con modo resonante TE_{10p} tendrá una longitud $p\lambda_{\mathrm{g}0}/2$, siendo p el número de oscilaciones longitudinales en módulo. Puesto que nuestro diseño parte de la subdivisión de los diferentes resonadores, que son aislados independientemente, es posible cortocircuitar los dos extremos de una guía donde propaga el modo TE_{10p} para conseguir así el *slope parameter* ideal que se tomará como punto de partida:

$$b_{\mathrm{ideal}} = \left(\frac{p\pi}{2}\right) Y_0 \left(\frac{\lambda_{\mathrm{g}_0}}{\lambda_0}\right)^2 \tag{3.2}$$

donde Y_0 correspone a la admitancia carcterística de la guía de onda, mientras que λ_0 y λ_{g_0} son la longitud de onda en espacio libre y en la guía a f_0.

El método clásico suele servirse de dos pasos para realizar la síntesis de los filtros de microondas:

1. Se ajusta el parámetro S_{11} de los elementos de acoplo con (2.18), buscando el valor de los inversores de admitancias obtenido anteriormente, siempre a la frecuencia de resonancia f_0.

2. Se selecciona la variable de diseño de las cavidades resonantes, teniendo en cuenta los desfases que introducen elementos de acoplo entre los que se encuentran. Para nuestro caso, este parámetro es representado por la longitud de cada resonador, que podrá ajustarse inicialmente mediante la expresión:

$$l_r = \frac{\lambda_{g_0}}{4\pi}(\phi_{11} + \phi_{22}) + \frac{n\lambda_{g_0}}{2} \tag{3.3}$$

   donde n es selecciona para conseguir la resonancia requerida en función del número de variaciones p longitudinales del modo resonante de la cavidad. En la expresió anterior, ϕ_{11} y ϕ_{22} son las fases del parámetro S_{11} y S_{22} del elemento de acoplo de entrada y de salida del resonador, respectivamente.

El actual problema de este método es que no se tienen en cuenta correctamente los elementos de acoplo adyacentes a los resonadores. Los elementos de acoplo modifican los campos electromagnéticos internos del resonador y alteran su comportamiento, variando asi su *slope parameter*, que teóricamente se suponía como un valor invariante. Ésto provoca la principal imprecisión en el método clásico, que deriva en un coste computacional de optimización elevado al partir de una estructura sintetizada con peores dimensiones iniciales. Esto nos lleva a proponer la forma de obtener el *slope parameter* descrita en el siguiente apartado.

3.1.2. *Slope parameter* con el nuevo método

Debido a la necesidad de establecer una equivalencia lo más rigurosa posible con el prototipo circuital y obtener así un "buen" *slope parameter* se propone el método desarrollado en la Sección 2.3. Tras el desarrollo teórico descrito en dicha sección se llega a la ecuación (2.20) que proporciona la reactancia del resonador cargado, de la que se extraerá el *slope parameter* mediante (2.22). Así, éste parámetro representará fielmente la variabilidad de en frecuencia de la susceptancia de los resonadores, pero ahora teniendo en cuenta los efectos de carga que le aplican los elementos de acoplo adyacentes.

Resulta muy interesante percatarse que, mediante este método, se consigue que los inversores de admitancia sean invariantes en frecuencia, debido a que este peso ha pasado a recaer sobre los resonadores adyacentes, propiciando una equivalencia perfecta con el prototipo circuital.

El cálculo del *slope parameter* utilizando éste método es la piedra angular de este Trabajo de Fin de Grado y, como se verá a continuación en la Sección 3.2, es el utilizado a partir de la 2ª iteración del *script* que se ha diseñado en *MATLAB*.

3.2. Aplicación

Teniendo en mente este nuevo método de obtención del *slope parameter* es posible implementarlo de forma que permita diseñar filtros en guía de onda en el que se disminuya substancialmente el coste computacional de la optimización.

Debido a la dependencia de los *slope parameters* con las dimensiones físicas de cada una de las estructuras y éstas, a su vez, depender de las constantes de inversión que se extraen a partir de dichos *slope parameters*, se ha implementado un proceso de diseño iterativo. Este proceso es el encargado de actualizar los parámetros en las distintas iteraciones y buscar la convergencía de los resultados. La convergencia de los resultados es conseguida una vez que el error entre las dimensiones físicas extraídas en dos iteraciones consecutivas sea inferior a un error mínimo preajustado por el diseñador. El *script* que implementa el método de diseño mediante el *slope parameter* funciona siguiendo el procedimiento ulterior:

- **1ª iteración:** al inicio del *script* de *MATLAB* se configuran las dimensiones fijas del filtro y se considera el *slope parameter* ideal de un resonador aislado usando la expresión (3.2). A partir de dicho valor, y mediante (2.12), se obtienen los valores iniciales de los inversores. Una vez obtenidos estos parámetros se procede a determinar la variable de diseño de cada elemento de acoplo mediante una optimización con *FEST3D*, hasta lograr la constante de inversión deseada de acuerdo con la ecuación (2.18). El último paso consiste en obtener el valor de la variable de diseño de cada resonador, que en nuestro caso será su longitud. Un excelente valor inicial se obtiene mediante (3.3), que luego se reajustará ligeramente hasta lograr que la frecuencia de resonancia del resonador cargado con sus elementos de acoplo de entrada y salida sea igual a la frecuencia central f_0 de la banda de paso del filtro.
- **2ª iteración:** tras la primera iteración y a partir de las dimensiones físicas del filtro obtenidas en ésta, se simula la respuesta de cada resonadore cargado con sus respectivos elementos de acoplo. Con ello obtenemos el parámetro S_{11}, que nos permite recalcular el valor del *slope parameter*, pero esta vez mediante el método propuesto en la Sección 2.3. El nuevo *slope parameter* obtenido para cada resonador representa un valor más fidedigno a la estructura real, y por tanto se utiliza para resintetizar los inversores que se habían calculado en la iteración anterior mediante la aplicación de (2.12). A partir de las nuevas constantes de inversión, se extraen unas dimensiones actualizadas para los elementos de acoplo y los resonadores siguiendo el mismo procedimiento empleado en la primera iteración.
- **Iteraciones superiores:** a partir de la segunda iteración (3ª y superiores) se vuelve a realizar el proceso seguido en la iteración 2 hasta conseguir la convergencia de los resultados. A modo de ejemplo, en la iteración N se calcularía el *slope parameter* real a partir de la estructura sintetizada en la iteración $N-1$ y, tras esto, se recalcularán unas nuevos valores de los inversores, y a continuación unos nuevas dimensiones físicas de los elementos de acoplo y resonadores reales.

El proceso se muestra de forma gráfica mediante un diagrama de flujo en la Figura 3.1.

El método iterativo descrito tiene infinidad de usos en el ambito de las microondas ya que se trata de un método general que parte de establecer una equivalencia lo más cercana posible al prototipo circuital. Dado que dicho prototipo circuital es la ”base” actual que sirve como punto de partida para el diseño de filtros de microondas da cabida a implementar los diseños en multitud de tecnologias.

En el caso de este Trabajo de Fin de Grado se pretende implementar el método iterativo descrito para la síntesis dimensional de un filtro en una tecnología concreta, los filtros de modo evanescente. Específicamente se implementaran los resonadores e inversores del prototípo circuital inicial mediante guías de onda *ridge* y guías rectangulares al corte, respectivamente.

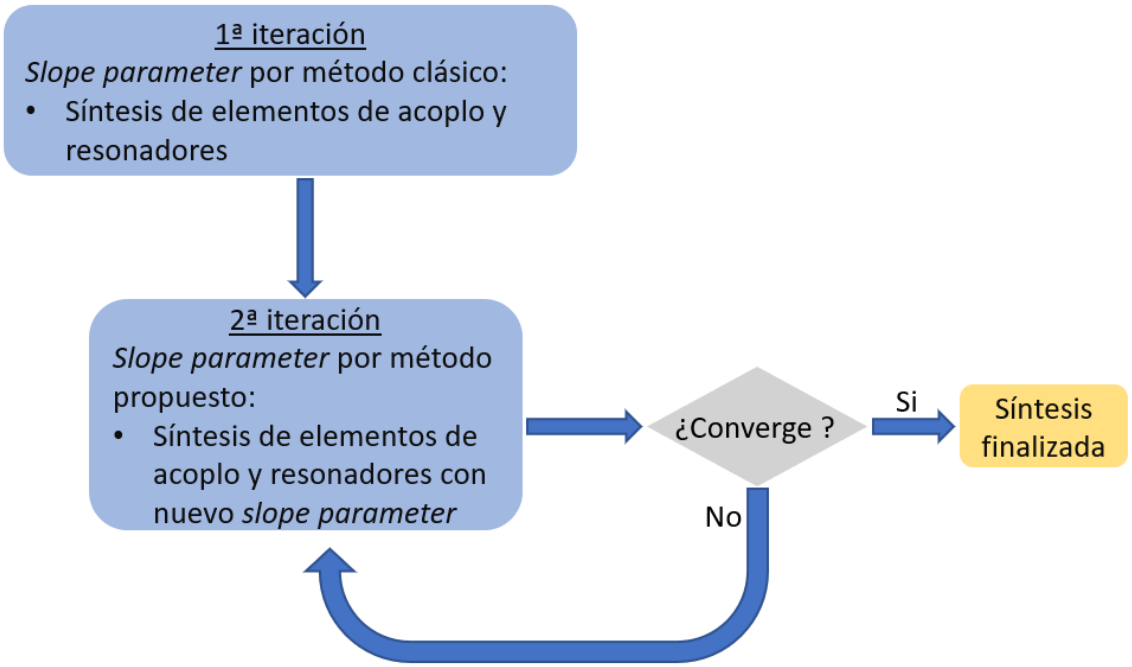


Figura 3.1: Proceso iterativo seguido por el método del *slope parameter*.

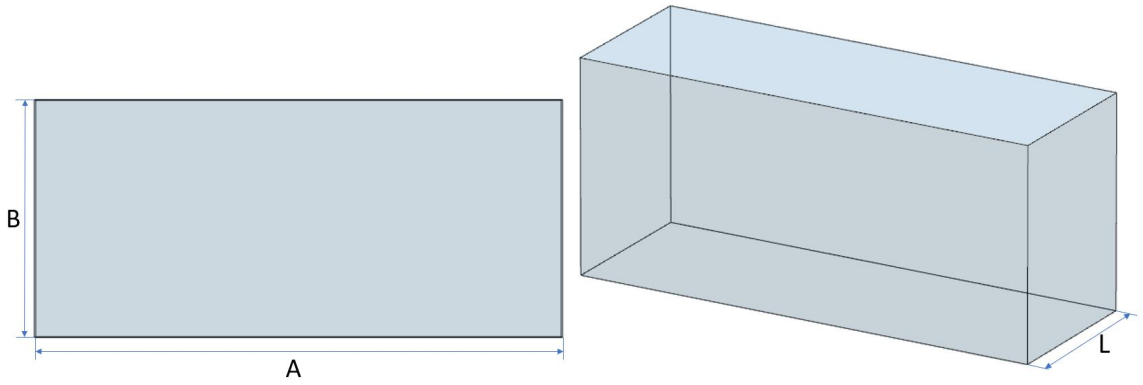
3.2.1. Elementos de un filtro de modo evanescente

En el caso que nos ocupa, como anteriormente se ha descrito, el filtro estará formado por dos elementos diferentes: las guías *ridge* y las guías *rectangulares* al corte. Las dimensiones de estos dos tipos de guías serán el objetivo final del *script*.

En las Figuras 3.2a y 3.2b se puede ver la topología de las guías *rectangulares*. Estas guías tendrán una anchura A que hagan que se encuentren al corte en la banda de frecuencias de trabajo, lo que permitirán usarlas en nuestro diseño para implementar los inversores de admitancia del prototipo que interconectan las diferentes cavidades resonantes, formadas a partir de las guías *ridge*.

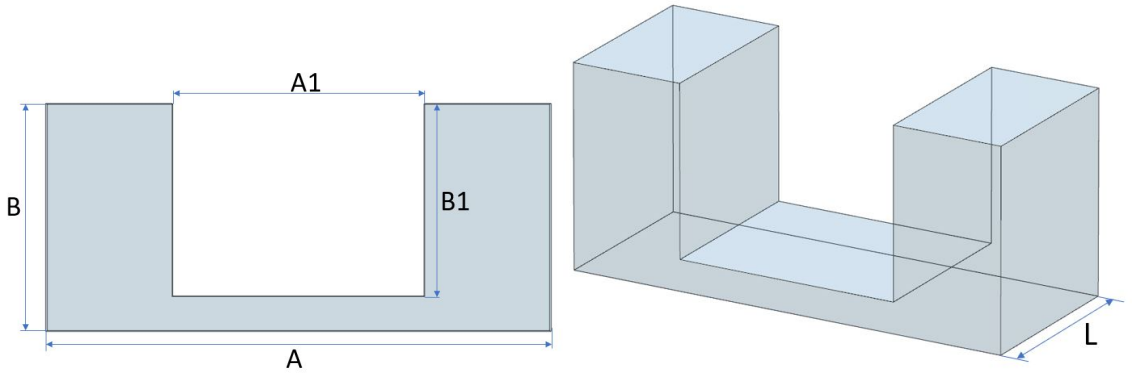
Las guías *ridge*, por su parte, incluyen en su sección superior un diente métalico que hace que pasen a estar en propagación. Por lo tanto, pueden hacer el papel de resonadores y, además, nos dan la ventaja de poder usarse como puertos de entrada al filtro y permiten también anchos de banda elevados. En el diseño de *MATLAB* se han parametrizado para permitir el diseño automatizado según las dimensiones de la guía que se vaya a utilizar. En las Figuras 3.3a y 3.3b se puede observar la topología de una de estas guías junto a sus dimensiones físicas más relevantes.

Obsérvese que en las Figuras 3.2a y 3.3a los parámetros de altura y anchura, A y B, serán los mismos en toda la estructura, dando lugar a un tubo de forma rectangular denominado *housing* en el que se insertarán los diferentes dientes que implementan las guías *ridge*. Esta es la implementación más habitual de los filtros de modo evanescente. La anchura A_1 y altura B_1 de dichos dientes serán las mismas en todas las guías *ridge*, y se elegirán para que éstas se encuentren comodamente en propagación en la banda de paso del componente. Por su parte, las variables de diseño serán las longitudes L de los dos tipos de elementos. En el caso de los tramos de guía rectangular al corte, un aumento de dicha longitud implica una reducción del acoplo entre los resonadores que interconecta.



(a) Sección transversal. **(b) Sección en perspectiva.**

Figura 3.2: Parámetros de la guía *rectangular*.



(a) Sección transversal. **(b) Sección en perspectiva.**

Figura 3.3: Parámetros de la guía *ridge*.

En el caso de las guías *ridge*, un aumento de la longitud L de lugar a un aumento de la capacidad en paralelo del resonador y por tanto una reducción de su frecuencia de resonancia. Por tanto, la longitud L nos permite controlar el parámetro fundamental que tiene que proporcionar cada elemento real para implementar su elemento correspondiente en el prototipo circuital equivalente.

3.3. Implementación

El método se ha diseñado para que, una vez se establezcan ciertas especificaciones del filtro a diseñar, esté todo completamente automatizado mediante *MATLAB*. La implementación se ha realizado utilizando el entorno de *MATLAB* como ”base” principal de trabajo. En él se especifican las características de ciertos aspectos de diseño que han de tenerse en cuenta, como por ejemplo: las frecuencias de corte superior e inferior, el orden del filtro (N), las pérdidas de retorno de la banda de paso (R_L), las medidas de las guías *ridge* y *rectangular*. *MATLAB* se encarga de realizar los cálculos teóricos para los que requieren estos datos y, posteriormente, crear los ficheros ”.fest3” de simulación y optimización.

Mediante el uso del CLI (del inglés *Command Line Interface*) que posee *FEST3D* se ha establecido una comunicación entre los dos softwares que permite utilizar el simulador electromagnético desde *MATLAB*. Esto permite realizar los calculos teóricos desde *MATLAB* y, desde el mismo entorno, obtener la respuesta de los componentes realizando una simulación y optimización llamando a *FEST3D* a través de la línea de comandos.

La creación de los archivos ”.fest3” se ha realizado con un conjunto de funciones que escriben diferentes ”bloques” de código en un fichero. Estos ”bloques” son los encargados de establecer, entre otros parámetros, las dimensiones de cada guía de onda, las características de la simulación, la optimización y los objetivos de esta última.

En total se han diseñado 6 funciones de *MATLAB* encargadas de crear los archivos ”.fest3” anteriormente descritos. Cada una de ellas tiene la función siguiente:

- **writeOpt:** crea el archivo y escribe el primer bloque del fichero, donde se establecen los parámetros de entrada que configuran la optimización como el nombre que tendrá el archivo ”.fest3”, el método de optimización número de parámetros a optimizar y los objetivos a los que estos han de llegar.
- **writeFest:** esta función se coloca siempre después de ”writeOpt”y escribe la configuración de la simulación que se va a realizar. Como parámetros de entrada más importantes tiene los límites de frecuencia de la simulación y el número de puntos en frecuencia donde se simularán.
- **writeRidge:** escribe el primer bloque de código referente a una guía de onda ridge. Como parámetros de entrada principales tiene: su numeración respecto a todos los elementos, su caracterización o no como puerto de entrada o salida y las dimensiones de la guía.
- **writeRectangular:** esta función desempeña el mismo trabajo que ”writeRidge” pero escribiendo el bloque de código referente a una guía de onda rectangular.
- **writeSteps:** es la encargada de establecer las discontinuidades que se forman entre dos elementos del filtro. Sus parametros de entrada son el número de elementos que forman el

diseño del fichero, la posición que ocupará la representación grafica de cada discontinuidad en la interfaz gráfica de *FEST3D* y como se interconectan las diferentes guías de la estructura entre sí.

Nótese que dichas funciones implementan un mayor número de variables de entrada, pero debido a el gran número de estas, se ha decidido exponer las consideradas como más importantes.

Al simularse los archivos ".fest3" se crean en el directorio las carpetas de simulación, que contienen los datos resultantes de la simulación. Puesto que algunos de ellos son necesarios para continuar el método, se han creado pequeñas funciones de apoyo que extraen dichos datos o realizan diferentes tareas como modificar ciertos datos de los archivos ".fest3", lanzar simulaciones, crear los archivos objetivo de simulación, obtener el *Slope parameter* real, etc.

En el apéndice A de este Trabajo de Fin de Grado se encuentran varias de las funciones desarrolladas. Se ha optado por no incluir el código fuente de todo el programa en *MATLAB*, al no haberse publicado aún el método de diseño desarrollado.

Capítulo 4

Resultados

Para visualizar como se ha implementado el método propuesto en *MATLAB* y como consigue abordar el diseños de filtros, se ha realizado el diseño de dos filtros de modo evanescente. Los resultados permiten percatarse de las mejoras en las estructuras sintetizadas (lo que implica menor esfuerzo computacional, al facilitar o incluso evitar la optimización electromagnética final), la baja cantidad de iteraciones necesarias para llegar a la convergencia en las dimensiones físicas y la capacidad de sintetizar estructuras con anchos de banda elevados.

4.1. Filtro de modo evanescente de 158 MHz de ancho de banda

El diseño se ha llevado a cabo buscando una banda de paso de 158 MHz de ancho comprendida entre $f_{c_1} = 5.921$ GHz y $f_{c_2} = 6.079$ GHz, es decir, centrada entorno a 6 GHz. Se requiere un filtro de orden $N = 4$, pérdidas de retorno $R_L = 25$ dB y que exhiba una función de transferencia de tipo Chebyshev. Las especificaciones de este filtro se han tomado del artículo [10], basado en la técnica de diseño por *slope parameter* utilizada en [8]. De esta forma es posible comparar los resultados de [10] con los obtenidos a partir del método propuesto en el presente Trabajo Fin de Grado.

Para el diseño se ha utilizado el mapeado en ω_0 clásico descrito en la Sección 2.2 para hacer la conversión del prototipo paso bajo a paso banda. Por lo general, este mapeado es válido para el ancho de banda que estamos considerando, generando a lo sumo un ligerísimo desplazamiento respecto a la frecuencia central de la banda.

El filtro se ha diseñado siguiendo la topología básica de un filtro de modo evanescente, dicho en otras palabras, utilizando como base una guía *rectangular* a la que se le añaden las inserciones metálicas en ciertas posiciones, formando así las secciones de guía *ridge*. La construcción se hace sobre una guía rectangular de dimensiones 12.5×5.625 mm^2, que está al corte en la frecuencia de trabajo. Las dimensiones de la sección transversal de la guia *ridge* se recogen en la tabla Tabla 4.1a, las cuales permiten la propagación de su modo fundamental en la banda de paso de interés.

Ya conocidas las especificaciones de la respuesta del filtro, y seleccionadas las dimensiones fijas de las guías, se puede proceder a ejecutar el primer paso de la técnica de sintésis (1ª iteración descrita en la Sección 3.2). Se comienza por obtener el *slope parameter* ideal de los resonadores mediante (3.2), considerando una cavidad resonante de longitud $\lambda_g/2$, es decir, con $p = 1$. Una

A	12.5
B	5.625
A1	6.25
B1	4.765

(a) Medidas de la guía *ridge*.

A	12.5
B	5.625

(b) Medidas de la guía *rectangular*.

Tabla 4.1: Dimensiones en mm **de las guías utilizadas (ver 3.3a y 3.2a).**

vez obtenido dicho *slope parameter* se procede a la síntesis de los inversores de admitancia mediante (2.12). El siguiente paso consiste en cálcular los elementos de acoplo adyacentes buscando la longitud correcta de los inversores que interconectan las guías *ridge*, l_{inv}. Este proceso se hace a través de una optimización con *FEST3D* que busca recuperar el valor de $|S_{11}|$ fijado al forzar la relación (2.18). Por último, se ajusta la longitud de cada resonador cargado con sus elementos de acoplo de entrada y salida para que resuene a la frecuencia central del filtro f_0.

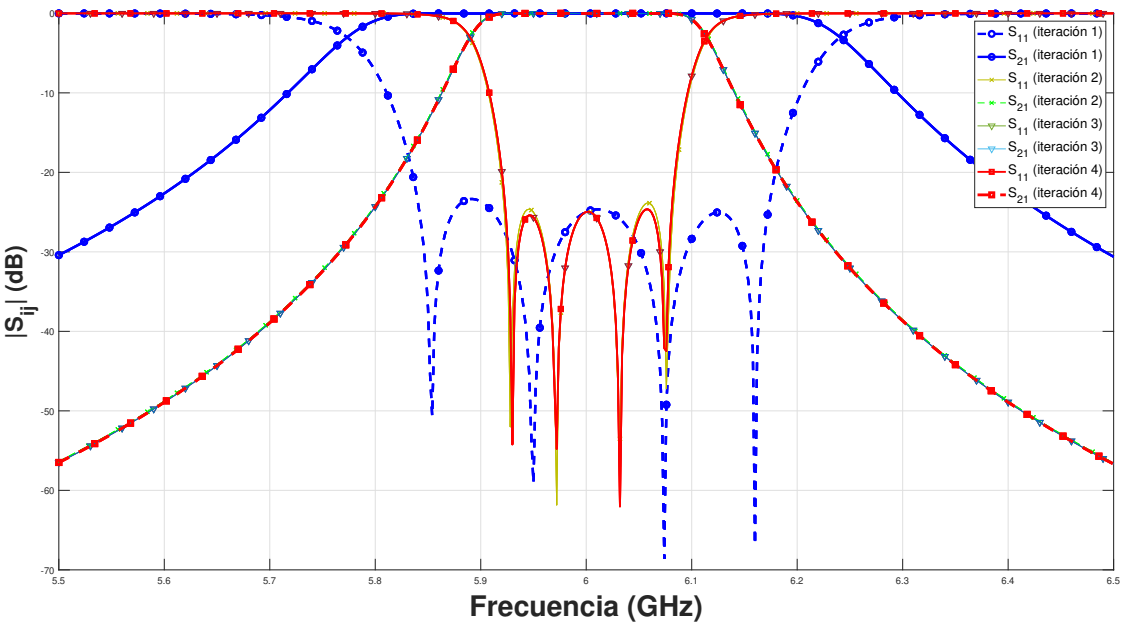


Figura 4.1: Respuesta del filtro respecto a cada iteración realizada con el nuevo método, obsérvese que a en la 2ª iteración casi se ha alcanzado la convergencia en la respuesta.

Tras la primera iteración, con todos los parámetros obtenidos mediante el procedimiento clásico (véase las Tablas 4.2 y 4.3), se obtiene la respuesta de la Figura 4.1 en color azul (señalada como 1ª iteración). Como se observa, el ancho obtenido sin optimización alguna es mucho mayor al requerido por las especificaciones dadas. En concreto se obtiene un ancho de banda de 340 MHz, lo que representa un 219 % del especificado, y se ha producido un aumento de 6 MHz sobre la frecuencia central de la banda.

Al continuar con el método pasamos a la 2ª iteración, y se realiza la primera de las iteraciones en la que se aplica el método propuesto en este Trabajo de Fin de Grado. Tras haber hecho la primera

síntesis dimensional en la 1ª iteración se procede a calcular el *slope parameter* real, que considerará los efectos de los elementos de acoplo adyacentes recién extraídos. De esta forma se corregirá el *slope parameter* utilizado en la 1ª iteración, que se trataba de un valor ideal. A continuación se recalcula con (2.12) los inversores de admitancia con el *slope parameter* corregido. Tras este paso se utiliza *FEST3D* para realizar una nueva optimización que permita ajustar la variable de diseño de los elementos de acoplo y resonadores, para implementar las constantes de inversión requeridas y fijar las frecuencias de resonancia de los resonadores cargados con dichos elementos de acoplo a f_0. Las nuevas longitudes de las guías del filtro se pueden ver en las Tablas 4.2 y 4.3 en la columna asociada a la iteración 2. Como se ve en la Figura 4.1 el ancho de banda del filtro ha mejorado enormemente y la respuesta está perfectamente centrada en frecuencia.

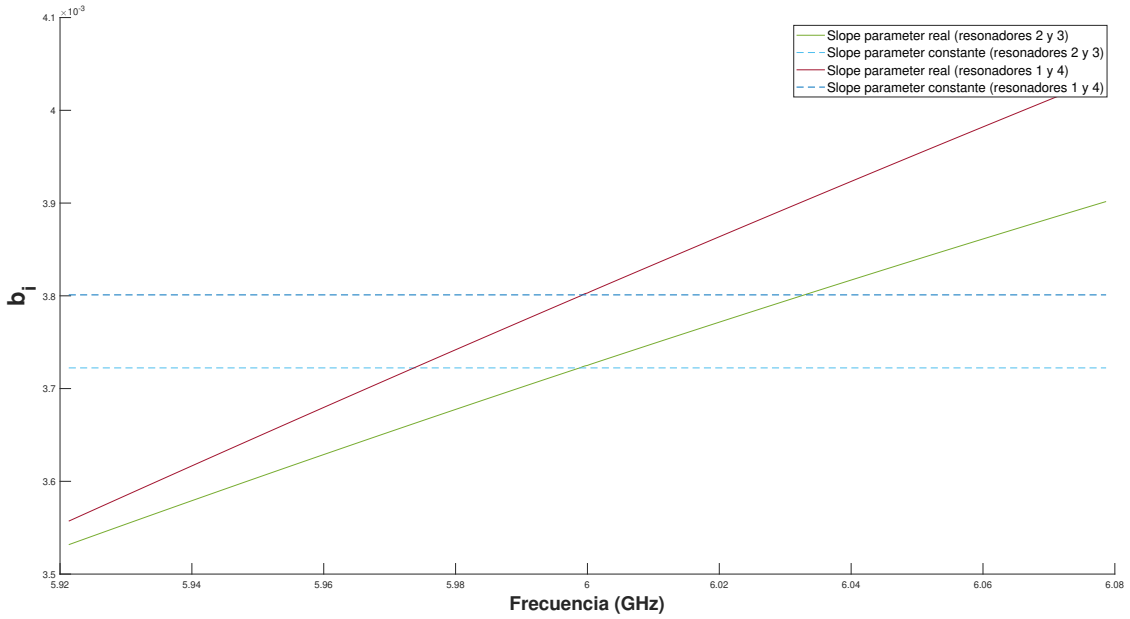


Figura 4.2: *Slope parameter* obtenido mediante el método propuesto.

La enorme mejora que se produce utilizando una única iteración se debe a la corrección hecha en los *slope parameter*, que se parecen mucho más a los *slope parameter* reales de los resonadores cargados de la estructura. En la Tabla 4.4 es posible ver como los *slope parameter* clásicos son más del doble que los *slope parameter* en la siguiente iteración, aumentando enormemente el error cuando se sintetizan los inversores con (2.12). En la Figura 4.2 se puede visualizar la variación en frecuencia de los *slope parameter* reales asociados a la derivada con la frecuencia de las susceptancias extraídas mediante (2.20), y los valor constantes del *slope parameter* que se han tomado (es decir, los asociados a la frecuencia f_0).

l_{inv}	Iteración 1	Iteración 2	Iteración 3	Iteración 4
$l_{01} = l_{45}$	3,2673	4,8610	4,8581	4,8581
$l_{12} = l_{34}$	7,9208	11,3214	11,3383	11,3382
l_{23}	9,2877	12,7217	12,7613	12,7614

Tabla 4.2: Evolución de la longitud en mm **de los inversores en cada iteración.**

l_{res}	Iteración 1	Iteración 2	Iteración 3	Iteración 4
$l_1 = l_4$	11,7060	10,4905	10,4914	10,4914
$l_2 = l_3$	9,7857	9,4369	9,4358	9,4358

Tabla 4.3: Evolución de la longitud en mm **de los resonadores en cada iteración.**

b_i	Iteración 1	Iteración 2	Iteración 3	Iteración 4
$b_1 = b_4$	0,0079	0,0038009	0,003801	0,003801
$b_2 = b_3$	0,0079	0,0037223	0,003722	0,003722

Tabla 4.4: *Slope parameter* en mS **a lo largo de las iteraciones.**

De esta forma, se continúa con las siguientes iteraciones, consiguiendo que las dimensiones físicas varíen por debajo de 1 micra (condición de convergencia) tras la 4ª iteración. Es interesante observar cómo entre la 1ª iteración (método clásico) y la 2ª iteración (método del *slope parameter*) se produce un cambio muy significativo tanto en el *slope parameter* como en los valores de longitud de las guías, mientras que en las siguientes iteraciones los cambios son cada vez más insignificantes debido a que se está alcanzando convergencia. De hecho, los *slope parameter* reales de los resonadores casi no tienen variación entre la 2ª iteración, la 3ª y la 4ª.

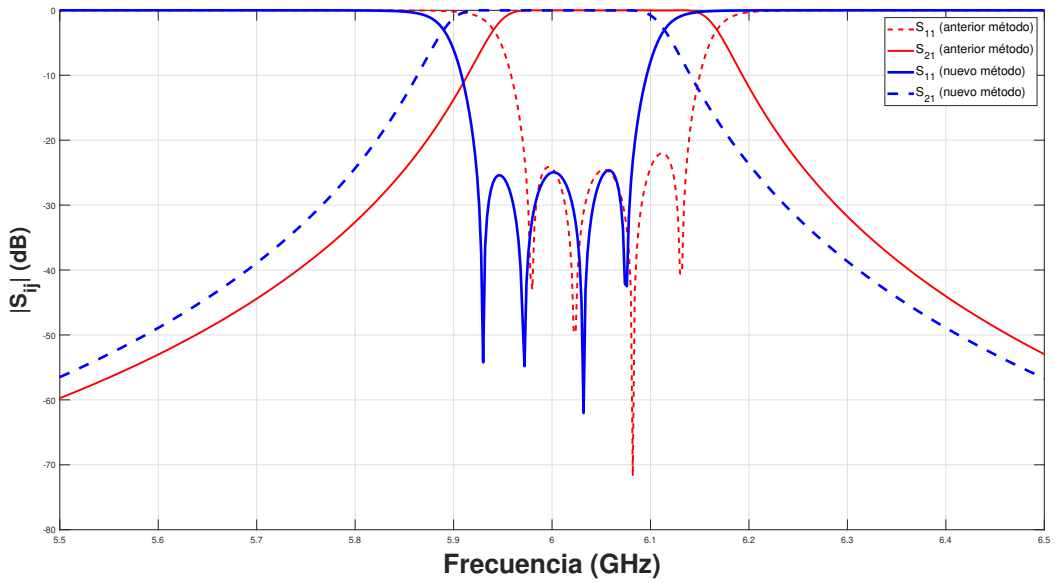


Figura 4.3: Respuesta del filtro de 158 MHz **mediante el método propuesto y el propuesto en [10]**

Puesto que el filtro esta diseñado respecto al considerado en [10], es posible evidenciar ciertas diferencias entre las respuestas de las estructuras sintetizadas por los dos métodos. En primer lugar se observa una deriva en frecuencia hacia arriba de 54 MHz y un ancho de banda ligeramente superior (163 MHz) al de las especificaciones en la respuesta de la estructura obtenida mediante [10]. En nuestro diseño estas ligeras variaciones no se producen gracias al uso del nuevo método

propuesto, revelando claramente que proporciona mejores resultados que la técnica más avanzada hasta la fecha. Así mismo, la obtención de un filtro con una respuesta prácticamente perfecta evita tener que realizar una optimización posterior (el paso más costoso computacionalmente), lo que aumenta drásticamente la eficiencia computacional del método.

Para este filtro se ha fijado un error mínimo de 1 µm, puesto que el error mínimo de fabricación por fresado de aluminio para un buen fabricante suele rondar entre los 5 µm y los 15 µm. De este modo, el error de diseño está bastante por debajo del error de fabricación.

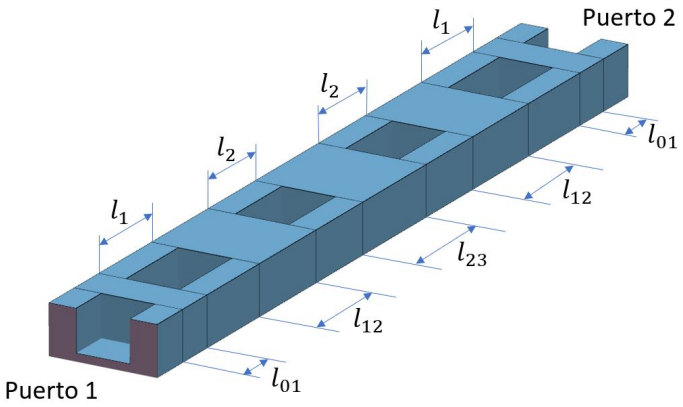


Figura 4.4: Vista tridimensional de la estructura en *FEST3D*, resultante de la 4ª iteración.

Respecto de las especificaciones propuestas y que el filtro ha de cumplir vemos que en la banda de paso el filtro tiene una perdidas de retorno mayores a 24.95 dB en la banda de paso, y se ajusta perfectamente a los requisitos de ancho de banda. Dicho esto, se puede concluir con que se ha hecho el modelo del filtro cumpliendo muy rigurosamente las especificaciones. La estructura diseñada puede visualizarse en la Figura 4.4.

4.2. Filtro de modo evanescente de 500 MHz de ancho de banda

El caso de este filtro es únicamente ilustrativo y no se encuentra en ningún otro articulo. El objetivo de este diseño es poder visualizar la validez de el método del *slope parameter* en filtros de elevado ancho de banda. El filtro en concreto se ha diseñado manteniendo la frecuencia central de 6 GHz del ejemplo anterior, pero aumentando su ancho de banda mediante unas frecuencias de corte superior e inferior de 6.251 MHz y 5.751 MHz, respectivamente. El ancho de banda relativo Δ es ahora del 8.33 %. Las perdidas de retorno en la banda de paso se han reducido a $R_{\mathrm{L}} = 23$ dB.

Tras introducir las especificaciones y lanzar la aplicación se consiguen las dimensiones observables en las Tablas 4.5, 4.6 y 4.7. El tiempo de ejecución aproximada del *script* de *MATLAB* es de 31 mín, en los que transcurren 7 iteraciones del método.

El proceso seguido para este filtro es el mismo, la unica diferencia apreciable es el número de iteraciones realizadas. A partir de la 6ª iteración la diferencia radica en décimas de micras para las dimensiones físicas y una variación inapreciable para el *slope parameter*.

En la Figura 4.5 se puede observar la respuesta de el filtro si se implementa con las dimensiones asociadas a su iteración. Como se observa en la primera iteración (correspondiente al método

l_{inv}	Iteración 1	Iteración 2	Iteración 3	Iteración 4	Iteración 5	Iteración 6	Iteración 7
$l_{01} = l_{45}$	1,06548	3,29654	2,55875	2,61299	2,60619	2,60703	2,60692
$l_{12} = l_{34}$	3,05527	7,02293	6,27328	6,32450	6,31762	6,31847	6,31835
l_{23}	4,27785	7,49198	6,27328	7,56451	7,56542	7,56530	7,56534

Tabla 4.5: Evolución de la longitud en mm **de los inversores en cada iteración.**

b_i	Iteración 1	Iteración 2	Iteración 3	Iteración 4	Iteración 5	Iteración 6	Iteración 7
$b_1 = b_4$	0,0026877	0,0038208	0,00372234	0,00373450	0,00373302	0,0037332	0,00373320
$b_1 = b_4$	0,0038688	0,0037949	0,00380754	0,00380677	0,00380687	0,0038068	0,00380687

Tabla 4.6: *Slope parameter* en mS **a lo largo de las iteraciones.**

l_{res}	Iteración 1	Iteración 2	Iteración 3	Iteración 4	Iteración 5	Iteración 6	Iteración 7
$l_1 = l_4$	15,3043	11,8718	12,6693	12,6076	12,6154	12,6144	12,6146
$l_2 = l_3$	13,0610	10,1642	10,3155	10,3048	10,3064	10,3062	10,3062

Tabla 4.7: Evolución de la longitud en mm **de los resonadores en cada iteración.**

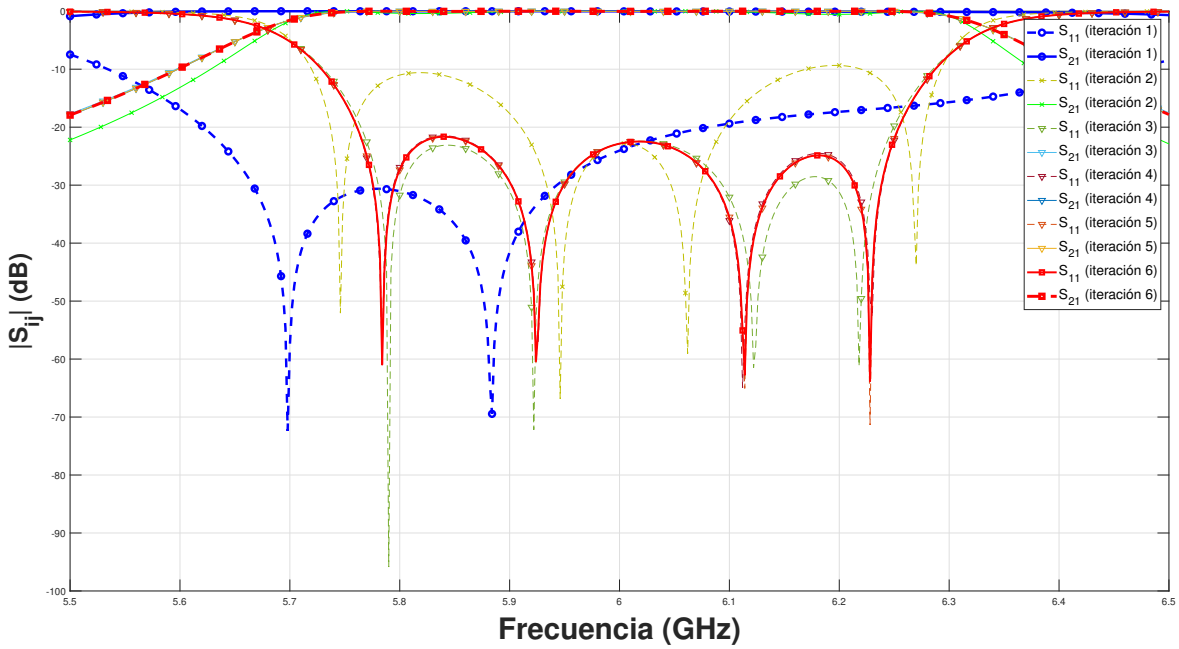


Figura 4.5: Respuesta del filtro respecto a cada iteración.

tradicional), debido a que el *slope parameter* ideal tiene un error muy grande llegan incluso a desaparecer dos de los 4 ceros de reflexión. Para este caso, el ancho de banda ha aumentado desmesuradamente y la frecuencia central se ha desplazado hacia abajo en frecuencia, siendo esta respuesta inaceptable.

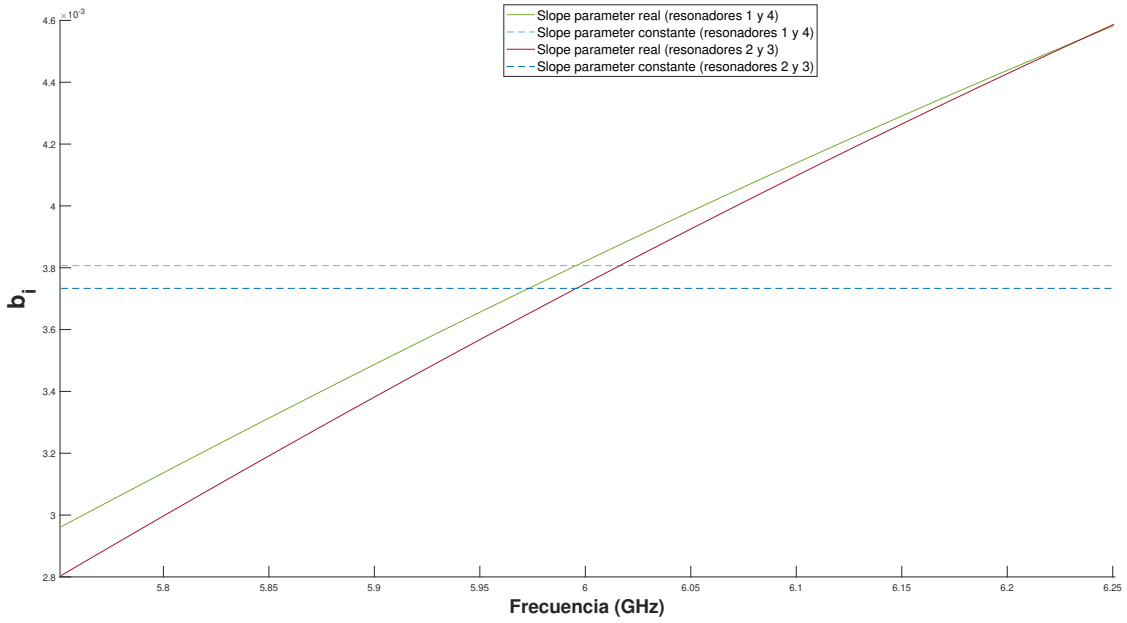


Figura 4.6: Evolución del *slope parameter* a lo largo del ancho de banda.

A partir de la 3ª iteración la respuesta no varía excesivamente, y poco a poco se va asentando hasta llegar a la deseada. El resultado final es un filtro con un desplazamiento hacia arriba en frecuencia de unos 16 MHz sobre la frecuencia central y un ancho de banda de 506 MHz (lo que corresponde a un error de 0.27 % y un 1.2 %, respectivamente). Respecto al requisito sobre la perdidas de retorno, únicamente se ha conseguido esta especificación para el tercero de los lóbulos del parámetro S_{11}, ya que el primer y segundo lóbulo se encuentran respectivamente 1.34 dB y 0.52 dB por encima de de la especificación de 23 dB. Este error es aceptable para el filtro, ya que las desviaciones por imprecisiones en la fabricación suelen ser mayores. Estos errores son debido al gran margen frecuencial en el que se trabaja, lo que implica una importante variación en el *slope parameter* (véase la Figura 4.6) en los margenes del ancho de banda y el que finalmente se utiliza en el método para obtener las dimensiones de la estructura.

Finalizando este diseño se puede extraer que se ha conseguido una estructura que, aunque no cumple estrictamente las especificaciones, está muy cerca, y ello conllevaría que con una simple optimización posterior permitiría cumplir las especificaciones por completo. Como última observación se puede añadir que empleando un mapeado paso bajo a paso banda diferente se podría mejorar la respuesta del filtro y disminuir, o incluso evitar, el desplazamiento sobre la frecuencia central y la variación sobre el ancho de banda. De todos modos para un mapeado clásico en ω_0 se ha demostrado que se tiene una buena respuesta a expensas de los leves errores anteriormente comentados.

Capítulo 5

Conclusiones

En este capítulo recogeremos tanto las conclusiones más significativas obtenidas en el presente Trabajo de Fin de Grado, como el cumplimiento del principal objetivo: el diseño de un *script* que permita implementar el método del *slope parameter* para el diseño de filtros de modo evanescente. En la Sección 5.1 hablaremos de las ventajas obtenidas con el novedoso método automatizado desarrollado en *MATLAB* y las facilidades que proporciona a los diseñadores, mientras que en la Sección 5.2 comentaremos los posibles futuras implementaciones de dicho método.

5.1. Presente

Como se ha visto en la Sección 4.1, la implementación del método del *slope parameter* permite acercar los diseños de estructuras de microondas a su prototípo circuital, aumentando la semejanza entre ambos, y permitiendo obtener dimensiones más cercanas a las óptimas que permitan cumplir las especificaciones planteadas. El uso de modernos simuladores y optimizadores electromagneticos junto al nuevo método, en nuestro caso *FEST3D*, permiten ”saltarse” las aproximaciones sobre las que se sostiene el método clásico y evitar así los errores que estas producen y que luego nos fuerzan a realizar costosas optimizaciones. La reducción de estos errores permite la obtención de estructuras iniciales de alta calidad que posibilitan evitar la optimización electromagnética de toda la estructura, o, en el peor de los casos, únicamente necesitar un rápido refine final. Este hecho nos permite disminuir enormemente los tiempos de optimización y simulación, como en el ejemplo de la Sección 4.1, que requiere de unos 15 mín en un ordenador de sobremesa convencional para completar el diseño.

La síntesis dimensional con el nuevo método del *slope parameter* propuesto evalúa el comportamiento frecuencial de los resonadores de las estructuras teniendo en cuenta los efectos de carga de los elementos de acoplo adyacentes. De esta forma, proporciona una vía por la que poder sintetizar los filtros con anchos de banda elevados.

Dicho método parte de las técnicas de diseño clásicas (correspondiente a la 1ª iteración descrita en la Sección 3.2) puesto que se necesitan unos parámetros de partida iniciales y, a partir de ellas se utiliza el *slope parameter* real para sintetizar las nuevas dimensiones de la estructura, que se van corrigiendo sucesivamente a cada iteración hasta llegar a una convergencia. Con todo, el presente y novedoso método se ha implementado únicamente para estructuras formadas por guías

ridge y *rectangulares* que formaran los filtros de modo evanescente, pero la nueva técnica es muy interesante para el diseño de que otros tipos de estructuras implementadas en cualquier tipo de tecnología.

El proceso desarrollado además está completamente automatizado, de forma que el usuario sólo ha de indicar las especificaciones y las dimensiones fijas de la estructura.

5.2. Futuro

Debido a que el presente método es una mejora de un aspecto básico del procedimiento clásico de diseño de los filtros de microondas, permite utilizase en innumerables estructuras y tecnologías. En vista de ello es posible abrir una nueva vía de investigación con el objetivo de implementar este novedoso método en, no solo las estructuras abordadas en este trabajo, sino a muchas otras.

Una vez sea posible desarrollar este método para diferentes tipos de estructuras de microondas en *MATLAB* y *FEST3D*, el destino final podría ser su integración en algun simulador electromagnético comercial, como podría ser *FEST3D*, *CST Studio Suite*, *ANSYS* etc.

En conclusión, se espera que el método *slope parameter* juegue un papel relevante a corto y largo plazo en el ámbito del desarrollo de estructuras de microondas.

Bibliografía

[1] Chandra M. Kudsia Richard J. Cameron y Raafat R. Mansour. *Microwave Filters for Communication Systems: Fundamentals, Design, and Applications*. Wiley Telecom, 2018, págs. 6-7. ISBN: 9781119292371. URL: https://ieeexplore.ieee.org/book/8341843.

[2] W. Cauer. *Die Verwirklichung der Wechselstromwiderst¨ande vorgeschriebener Frequenzabh¨angigkeit*. 1926, págs. 355-388.

[3] S.Darlington. *Synthesis of reactance 4-poles which produce prescribed insertion loss characteristics*. Journal of Mathematics y Physics, 1939, págs. 257-353.

[4] Seymour B. Cohn. *Direct-Coupled-Resonator Filters*. IRE, 1957, pág. 188. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4056485.

[5] R.V. Snyder G. Matthaei y R. Levy. "Design of microwave filters". En: (mar. de 2002), págs. 783-793. ISSN: 1557-9670. DOI: 10.1109/22.989962. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=989962.

[6] Santiago Cogollos y Ana Vidal. *Comunicaciones Espaciales*. UPV, 2019, págs. 6-7.

[7] Ralph Levy y Seymour B. Cohn. *A History of Microwave Filter Research, Design, and Development*. Vol. 32. IEEE, sep. de 1984, pág. 1056. ISBN: 1557-9670. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1132817.

[8] H. Y. Hwang y S.-W. Yun. "The design of bandpass filters considering frequency dependence of inverters". En: *Microwave Journal* 45.9 (2002), págs. 154-163.

[9] F. M. Vanin, D. Schmitt y R. Levy. "Dimensional synthesis for wide-band waveguide filters and diplexers". En: *IEEE Transactions on Microwave Theory and Techniques* 52.11 (2004), págs. 2488-2495.

[10] V. Tornielli y F. de Paolis. "Dimensional synthesis of evanescent-mode ridge waveguide bandpass filters". En: *IEEE Transactions on Microwave Theory and Techniques* 66.2 (2018), págs. 954-961.

[11] Santiago Cogollos Borrás. *Fundamentos de la teoría de filtros*. Spanish. OCLC: 984821694. Valencia: Editorial Universitat Politècnica de València, 2016. ISBN: 978-84-9048-443-2.

Parte II

Anexos

Apéndice A

Códigos de *MATLAB*

En esta sección se pueden visualizar los códigos de las funciones utilizadas para crear los archivos ".fest3" descritos en la Sección 3.3.

A.1. Función *writeOpt*

```
function writeOpt(file_name,opt_method,...
                  formula,  file_line, ...
                  initial_size, tolerance, max_iterations,
                     target_error,...
                  number_parameters,...
                  varargin)
%Example of use ------> writeOpt('filtro','simplex',...
%                                 [4.85806585747373
   10.4913864628323 11.3382132626084],...
%                                 0.05, 1.0E-4, 1000, '1.0E
   -14',...
%                                 3, ...
%                                 'GoalInv_1');
%Input arguments:
%  - file_name:              name of the destination file
%  -opt_method:              optimization type (accepts '
   simplex' or 'powell')
%  - formula:                value of optimization
%  - initial_size:           initial size
%  - tolerance:              tolerance level
%  - max_iterations:         maximum number of iterations
%  - target_error:           target orror the optimizations
   looks for
%  - number_parameters:      number of parameters we'll
   optimize
%  - OptGoal:                target file of optimization
```

```
21
22
23 %%Index:
24 % 1. Check if file_name file exists
                                                    (DONE)
25 % 2. Check if varargin exists and set OptGoal
                                              (DONE)
26 % 3. Create data with input variables
                                                  (DONE)
27 % 4. Write Block code into "file_name" file
                                              (DONE)
28
29
30 %%1. Check if file_name file exists
31 %filename    = ['.\out\' file_name '.txt'];
32 filename    = [file_name '.fest3'];
33 fileID      = fopen(filename,'w');
34
35 %%2. Check existance of Optimization file at the end of the
   inputs
36 if nargin > 9
37     OptGoal = varargin{1};
38 end
39
40 %%3. Create data with input variables
41 C = cell(1,1);
42 C{1,1}  = ['begin "opt-2020"'];
43 C{end+1,1}  = ['  begin "general"'];
44
45 if strcmp(opt_method,'simplex') == 1              %opt method
   selected 'SIMPLEX'
46 C{end+1,1}  = ['    begin "simplex"'];
47 C{end+1,1}  = ['      initial_size          ' num2str(
   initial_size)];
48 C{end+1,1}  = ['    end "simplex"'];
49 else                                              %opt method
   selected 'POWELL'
50 C{end+1,1}  = ['    begin "powell"'];  %aqui podria poner
   variable para
51 C{end+1,1}  = ['      initial_size          ' num2str(
   initial_size)];%----------
52 C{end+1,1}  = ['      tolerance             ' num2str(tolerance)
   ];%----------
53 C{end+1,1}  = ['    end "powell"'];
54 end
55
```

```
56 C{end+1,1}  = ['    max_iterations           ' num2str(
       max_iterations)];
57 C{end+1,1}  = ['    target_error            ' num2str(
       target_error)];
58 C{end+1,1}  = ['  end "general"'];
59 C{end+1,1}  = ['  begin "parameters"'];
60 C{end+1,1}  = ['    number_parameters       ' num2str(
       number_parameters)];
61
62 for i = 1:number_parameters
63 clear aux;
64 aux{1,1}       = ['    begin "parameter"'];
65 aux{end+1,1}   = ['      name      "x' num2str(i) '"'];
66 aux{end+1,1}   = ['      file_line              ' num2str(
       file_line(i))];    %Aqui habria que ir contando
67 aux{end+1,1}   = ['      file_keyword           "1"'];    %las
       lineas que hay en el
68 aux{end+1,1}   = ['      enabled                "true"']; %
       codigo total
69 aux{end+1,1}   = ['      formula                "' num2str(
       formula(i),15) '"'];%se peude crear vector con diferentes
       valores
70 aux{end+1,1}   = ['      distribution           gaussian'];
71 aux{end+1,1}   = ['      gaussian_sigma         0.01'];
72 aux{end+1,1}   = ['    end "parameter"'];
73 C = vertcat(C,aux);
74 end
75
76 C{end+1,1}  = ['  end "parameters"'];
77 C{end+1,1}  = ['  begin "constraints"'];
78 C{end+1,1}  = ['    number_constraints      0'];
79 C{end+1,1}  = ['  end "constraints"'];
80 C{end+1,1}  = ['  begin "targets"'];
81
82 if strcmp(opt_method,'simplex') == 1
83 C{end+1,1}  = ['    number_targets          2 '];
84 C{end+1,1}  = ['    begin "target"'];
85 C{end+1,1}  = ['      engine_column_x       1'];
86 C{end+1,1}  = ['      engine_column_y       2'];
87 C{end+1,1}  = ['      target_file           ""'];
88 C{end+1,1}  = ['      target_column_x       1 '];
89 C{end+1,1}  = ['      target_column_y       2 '];
90 C{end+1,1}  = ['      key       "<=" '];
91 C{end+1,1}  = ['      weight                1.0 '];
92 C{end+1,1}  = ['      gui_discrete          "true" '];
93 C{end+1,1}  = ['      begin "AFS" '];
94 C{end+1,1}  = ['        error_stop          0.001 '];
```

```
95  C{end+1,1}  = ['         max_iteration       0 '];
96  C{end+1,1}  = ['         nweights            4 '];
97  C{end+1,1}  = ['         begin "weights" '];
98  C{end+1,1}  = ['           weight            0.25 '];
99  C{end+1,1}  = ['           weight            -0.25 '];
100 C{end+1,1}  = ['           weight            -0.25 '];
101 C{end+1,1}  = ['           weight            0.25 '];
102 C{end+1,1}  = ['         end "weights" '];
103 C{end+1,1}  = ['       end "AFS" '];
104 C{end+1,1}  = ['     end "target" '];
105 C{end+1,1}  = ['     begin "target" '];
106 C{end+1,1}  = ['       engine_column_x       1 '];
107 C{end+1,1}  = ['       engine_column_y       8 '];
108 C{end+1,1}  = ['       target_file           "" '];
109 C{end+1,1}  = ['       target_column_x       1 '];
110 C{end+1,1}  = ['       target_column_y       2 '];
111 C{end+1,1}  = ['       key         "<=" '];
112 C{end+1,1}  = ['       weight                0.3 '];
113 C{end+1,1}  = ['       gui_discrete          "true" '];
114 C{end+1,1}  = ['       begin "AFS" '];
115 C{end+1,1}  = ['         error_stop          0.001 '];
116 C{end+1,1}  = ['         max_iteration       0 '];
117 C{end+1,1}  = ['         nweights            4 '];
118 C{end+1,1}  = ['         begin "weights" '];
119 C{end+1,1}  = ['           weight            0.25 '];
120 C{end+1,1}  = ['           weight            -0.25 '];
121 C{end+1,1}  = ['           weight            -0.25 '];
122 C{end+1,1}  = ['           weight            0.25 '];
123 C{end+1,1}  = ['         end "weights" '];
124 C{end+1,1}  = ['       end "AFS" '];
125 C{end+1,1}  = ['     end "target" '];
126 else
127 C{end+1,1}  = ['     number_targets          1 '];
128 C{end+1,1}  = ['     begin "target" '];
129 C{end+1,1}  = ['       engine_column_x       1 '];
130 C{end+1,1}  = ['       engine_column_y       2 '];
131 C{end+1,1}  = ['       target_file           "' OptGoal '.out"'];
132 C{end+1,1}  = ['       target_column_x       1'];
133 C{end+1,1}  = ['       target_column_y       2'];
134 C{end+1,1}  = ['       key         "=" '];
135 C{end+1,1}  = ['       weight                1.0'];
136 C{end+1,1}  = ['       gui_discrete          "true"'];
137 C{end+1,1}  = ['       begin "AFS"'];
138 C{end+1,1}  = ['         error_stop          0.001'];
139 C{end+1,1}  = ['         max_iteration       0'];
140 C{end+1,1}  = ['         nweights            4'];
141 C{end+1,1}  = ['         begin "weights"'];
```

```
142 C{end+1,1}  = ['           weight              0.25'];
143 C{end+1,1}  = ['           weight             -0.25'];
144 C{end+1,1}  = ['           weight             -0.25'];
145 C{end+1,1}  = ['           weight              0.25'];
146 C{end+1,1}  = ['        end "weights"'];
147 C{end+1,1}  = ['      end "AFS"'];
148 C{end+1,1}  = ['    end "target"'];
149 end
150 C{end+1,1}  = ['  end "targets"'];
151 C{end+1,1}  = ['end "opt-2020"'];
152
153 %%4. Write RIDGE code into "file_name" file
154 %Write code. If file_name already exists its written in the
       final line
155 fprintf(fileID,'%s \n%s \n%s \n',C{:,1});
156 fclose(fileID);
157 end
```

Código A.1: Función "writeOpt"

A.2. Función *writeFest20*

```
1  function writeFest20(file_name,...
2                       start_freq, end_freq, number_points,...
3                       access_modes, basis_func, green_func,
                           taylor_expans)
4  %Example of use ------>  writeFest20('filtro',...
5  %                                    1, 5.99947989412416,
       5.99947989412416, 1,...
6  %                                    25,200,1500,4);
7  %Input arguments:
8  %   - file_name:                 name of the destination file
9  %   - start_freq:                starting simulation frequency
10 %   - end_freq:                  ending   simulation frequency
11 %   - number_points:             number of points in the
       simulation
12 %   - access_modes:              number of accessible modes
13 %   - basis_func:                number of basis functions
14 %   - green_func:                number of green functions?
15 %   - taylor_expans:             number of Taylor term series
16
17 %%Index:
18 %1. Check if file_name file exists
                                                    (DONE)
19 %2. Create data with input variables
                                                  (DONE)
```

```
20 %3. Write Block code into "file_name" file
                                        (DONE)
21
22 %%1. Check if file_name file exists
23 %filename    = ['.\out\' file_name '.txt'];
24 filename    = [file_name '.fest3'];
25 fileID      = fopen(filename,'a+');
26
27 %%2. Create data with input variables
28 C = cell(1,1);
29 C{1,1}      = ['begin "fest-2020"'];
30 C{2,1}      = ['  begin "general"'];
31 C{3,1}      = ['    begin "frequency_sweep"'];
32 C{4,1}      = ['      enable                "true"'];
33 C{5,1}      = ['      byOPT                   "false"'];
34 C{6,1}      = ['      disabled_byOPT        "false"'];
35 C{7,1}      = ['      discrete              "true"'];
36 C{8,1}      = ['      begin "AFS"'];
37 C{9,1}      = ['        error_stop          0.001'];
38 C{10,1}     = ['        max_iteration       ' num2str(
   number_points)];    %max_iterations
39 C{11,1}     = ['        nweights            4'];
40 C{end+1,1}  = ['        begin "weights"'];
41 C{end+1,1}  = ['          weight            0.25'];
42 C{end+1,1}  = ['          weight            -0.25'];
43 C{end+1,1}  = ['          weight            -0.25'];
44 C{end+1,1}  = ['          weight            0.25'];
45 C{end+1,1}  = ['        end "weights"'];
46 C{end+1,1}  = ['      end "AFS"'];
47 C{end+1,1}  = ['      start      ' num2str(start_freq,15)];
   %start_freq
48 C{end+1,1}  = ['      end        ' num2str(end_freq,15)];    %
   end_freq
49 C{end+1,1}  = ['      number_points         ' num2str(
   number_points)]; %number_points
50 C{end+1,1}  = ['    end "frequency_sweep"'];
51 C{end+1,1} = ['    theta       0.0'];
52 C{end+1,1} = ['    phi         0.0'];
53 C{end+1,1} = ['    metric_unit                    "mm"'];
54 C{end+1,1} = ['    begin "gui"'];
55 C{end+1,1} = ['      dielectric_permittivity          1.0'];
56 C{end+1,1} = ['      dielectric_permeability          1.0'];
57 C{end+1,1} = ['      dielectric_conductivity          0.0'];
58 C{end+1,1} = ['      metal_resistivity                0.0'];
59 C{end+1,1} = ['      number_accessible_modes          ' num2str
   (access_modes)];    %access_modes
```

```
60  C{end+1,1} = ['      number_basis_functions              ' num2str
        (basis_func)];      % basis_func
61  C{end+1,1} = ['      number_modes_green_function         ' num2str
        (green_func)];      % green_func
62  C{end+1,1} = ['      number_terms_taylor_expansion       ' num2str
        (taylor_expans)];   % taylor_expans
63  C{end+1,1} = ['      reference_start_plot_3d             1'];
64  C{end+1,1} = ['    end "gui"'];
65  C{end+1,1} = ['   begin "symmetries"'];
66  C{end+1,1} = ['      x          '];
67  C{end+1,1} = ['    end "symmetries"'];
68  C{end+1,1} = ['    begin "reference"'];
69  C{end+1,1} = ['      origin                               1'];
70  C{end+1,1} = ['    end "reference"'];
71  C{end+1,1} = ['    begin "multipactor"'];
72  C{end+1,1} = ['      char_length_multipactor              1.0'];
73  C{end+1,1} = ['      mode                                 "
        single_1freq"'];
74  C{end+1,1} = ['      frequency                            9.0'];
75  C{end+1,1} = ['      begin "frequency_sweep"'];
76  C{end+1,1} = ['        start                 9.0'];
77  C{end+1,1} = ['        end                   15.0'];
78  C{end+1,1} = ['        number_points         3'];
79  C{end+1,1} = ['      end "frequency_sweep"'];
80  C{end+1,1} = ['      begin "multicarrier_sweep"'];
81  C{end+1,1} = ['        number_of_carriers                 0'];
82  C{end+1,1} = ['      end "multicarrier_sweep"'];
83  C{end+1,1} = ['    end "multipactor"'];
84  C{end+1,1} = ['    begin "corona"'];
85  C{end+1,1} = ['      char_length_corona                   1.0'];
86  C{end+1,1} = ['      mode
                  "single_1freq"'];
87  C{end+1,1} = ['      frequency
          0.0'];
88  C{end+1,1} = ['      begin "frequency_sweep"'];
89  C{end+1,1} = ['        start                 9.0'];
90  C{end+1,1} = ['        end                   15.0'];
91  C{end+1,1} = ['        number_points         3'];
92  C{end+1,1} = ['      end "frequency_sweep"'];
93  C{end+1,1} = ['    end "corona"'];
94  C{end+1,1} = ['    begin "emfield"'];
95  C{end+1,1} = ['      char_length_emfield                  1.0'];
96  C{end+1,1} = ['    end "emfield"'];
97  C{end+1,1} = ['  end "general"'];
98  C{end+1,1} = ['  begin "waveguides"'];
99
100 %%4. Write fest20 block code into "file_name" file
```

```
101 %Write code. If file_name already exists its written in the
        final line
102 fprintf(fileID,'%s \n%s \n%s \n',C{:,1});
103 fclose(fileID);
104 end
```

Código A.2: Función "writeFest20"

A.3. Función *writeRidge*

```
1  function writeRidge(file_name, WG_num,isport, GUI_x_pos, ...
2                      a,b,a1,b1,length, ...
3                      precision, rect_modes, ...
4                      access_modes, basis_func, green_func,
                           taylor_expans)
5  %Example of use ------>  writeRidge('filtro', 3,2, 280, ...
6  %                                    12.5,5.625,6.25,4.765,5, ...
7  %                                    5, 1500, ...
8  %                                    25, 200, 1500, 4)
9  %Input arguments:
10 %   - file_name:                  name of the destination file
11 %   - WG_num:                     numeration onf the waveguide
12 %   - isport:                     if its a tl has a value of 0
13 %                                   if is a port the number of the
       port
14 %   - GUI_x_pos:                  X axis of the block in FEST3D
       GUI
15 %   - a:                          waveguide width
16 %   - b:                          waveguide height
17 %   - a1:                         steel insertion width
18 %   - b1:                         steel insertion heigth
19 %   - long:                       waveguide length
20 %   - precision:                  precision of the wg
21 %   - rect_modes:                 number of rectangular modes in
       the box
22 %   - access_modes:               number of accessible modes
23 %   - basis_func:                 number of basis functions
24 %   - green_func:                 number of green functions?
25 %   - taylor_expans:              number of Taylor term seriesç
26
27 %%Index:
28 %1. Check if file_name file exists
                                                    (DONE)
29 %2. Create data with input variables
                                                  (DONE)
```

```
30 % 3. Check if its a waveguide port
                                                    (DONE)
31 % 4. Write Block code into "file_name" file
                                                (DONE)
32
33 %%1. Check if file_name file exists
34 %filename    = ['.\out\' file_name '.txt'];
35 filename    = [file_name '.fest3'];
36 fileID = fopen(filename,'a+');
37
38 %%2. Create data with input variables
39 C{1,1}  = ['    begin "ridge"'];
40 C{2,1}  = ['      number                ' num2str(WG_num)];
41 C{3,1}  = ['      begin "gui"'];
42 C{4,1}  = ['        x_pos               ' num2str(GUI_x_pos)];
43 C{5,1}  = ['        y_pos               140'];
44 C{6,1}  = ['      end "gui"'];
45 C{7,1}  = ['      begin "multipactor"'];
46 C{8,1}  = ['        analize
   "false"'];
47 C{9,1}  = ['        use_general_settings          "true"'];
48 C{10,1} = ['      end "multipactor"'];
49 C{11,1} = ['      begin "corona"'];
50 C{12,1} = ['        analize
   "false"'];
51 C{13,1} = ['        use_general_settings          "true"'];
52 C{14,1} = ['      end "corona"'];
53 C{15,1} = ['      number_rectangular_modes      ' num2str(
   rect_modes)];
54 C{16,1} = ['      precision
   ' num2str(precision)];
55 C{17,1} = ['      a             ' num2str(a,15)];
56 C{18,1} = ['      b             ' num2str(b,15)];
57 C{19,1} = ['      aa            0.0'];
58 C{20,1} = ['      bb            0.0'];
59 C{21,1} = ['      a1            ' num2str(a1,15)];
60 C{22,1} = ['      b1            ' num2str(b1,15)];
61 C{23,1} = ['      a2            0.0'];
62 C{24,1} = ['      b2            0.0'];
63 C{25,1} = ['      rext          0.0'];
64 C{26,1} = ['      rint          0.0'];
65 C{27,1} = ['      x0            0.0'];
66 C{28,1} = ['      y0            0.0'];
67 C{29,1} = ['      alpha         0.0'];
68 C{30,1} = ['      l             ' num2str(length,15)];
69 C{31,1} = ['      dielectric_permittivity       1.0'];
70 C{32,1} = ['      dielectric_permeability       1.0'];
```

```
71 C{33,1} = ['       dielectric_conductivity              0.0'];
72 C{34,1} = ['       metal_resistivity
       0.0'];
73 C{35,1} = ['       number_accessible_modes             ' num2str(
       access_modes)];
74 C{36,1} = ['       number_basis_functions              ' num2str(
       basis_func)];
75 C{37,1} = ['       number_modes_green_function         ' num2str(
       green_func)];
76 C{38,1} = ['       number_terms_taylor_expansion       ' num2str(
       taylor_expans)];
77 C{39,1} = ['       begin "subtype"'];
78
79 %%3. Check if its a waveguide port
80 if isport == 0
81     C{40,1} = ['         begin "tl"'];
82     C{41,1} = ['         end "tl" '];
83     C{42,1} = ['       end "subtype"'];
84     C{43,1} = ['       begin "emfield"'];
85     C{44,1} = ['         char_length_emfield                1.0'];
86     C{45,1} = ['       end "emfield"'];
87     C{46,1} = ['     end "ridge"'];
88 else             %We must write all code from here because we
       dont know exactly the col number
89     C{40,1} = ['         begin "ioport"'];
```

Código A.3: Función "writeRidge"

A.4. Función *writeRectangular*

```
1  function writeRectangular(file_name, WG_nm,isport, GUI_x_pos, ...
2                    a,b,length, ...
3                    access_modes, basis_func, green_func,
                        taylor_expans)
4  %Example of use ------>  writeRectangular('filtro', 3, 2, 280,
       ...
5  %                                          12.5, 5.625,
       8.35062431433568, ...
6  %                                          5, 1500, ...
7  %                                          25, 200, 1500, 4)
8  %Input arguments:
9  %   - file_name:                 name of the destination file
10 %   - WG_nm:                     numeration onf the waveguide
11 %   - isport:                    if its a tl has a value of 0
12 %                                  if is a port the number of the
       port
```

```
13 %  - GUI_x_pos:                  X axis of the block in FEST3D
   GUI
14 %  - a:                          waveguide width
15 %  - b:                          waveguide height
16 %  - long:                       waveguide length
17 %  - precision:                  precision of the wg
18 %  - rect_modes:                 number of rectangular modes in
   the box
19 %  - access_modes:               number of accessible modes
20 %  - basis_func:                 number of basis functions
21 %  - green_func:                 number of green functions?
22 %  - taylor_expans:              number of Taylor term series
23 %
24
25 %%Index:
26 %1. Check if file_name file exists
                                                        (DONE)
27 %2. Create data with input variables
                                                      (DONE)
28 %3. Check if its a waveguide port
                                                          (DONE)
29 %4. Write rectangular WG code into "file_name" file
                                  (DONE)
30
31 %%1. Check if file_name file exists
32 %filename    = ['.\out\' file_name '.txt'];
33 filename    = [file_name '.fest3'];
34 fileID = fopen(filename,'a+');
35
36 %%2. Create data with input variables
37 C{1,1}  = ['    begin "rectangular"'];
38 C{2,1}  = ['      number                ' num2str(WG_num)];
39 C{3,1}  = ['      begin "gui"'];
40 C{4,1}  = ['        x_pos               ' num2str(GUI_x_pos)];
41 C{5,1}  = ['        y_pos               140'];
42 C{6,1}  = ['      end "gui"'];
43 C{7,1}  = ['      begin "multipactor"'];
44 C{8,1}  = ['        analize
   "false"'];
45 C{9,1}  = ['        use_general_settings            "true"'];
46 C{10,1} = ['      end "multipactor"'];
47 C{11,1} = ['      begin "corona"'];
48 C{12,1} = ['        analize
   "false"'];
49 C{13,1} = ['        use_general_settings            "true"'];
50 C{14,1} = ['      end "corona"'];
51 C{15,1} = ['      a           ' num2str(a,15)]; %--------
```

```
52 C{16,1} = ['      b          ' num2str(b,15)]; %----------
53 C{17,1} = ['      l          ' num2str(length,15)]; %----------
54 C{18,1} = ['      dielectric_permittivity            1.0'];
55 C{19,1} = ['      dielectric_permeability            1.0'];
56 C{20,1} = ['      dielectric_conductivity            0.0'];
57 C{21,1} = ['      metal_resistivity
       0.0'];
58 C{22,1} = ['      number_accessible_modes            ' num2str(
       access_modes)];
59 C{23,1} = ['      number_basis_functions             ' num2str(
       basis_func)];
60 C{24,1} = ['      number_modes_green_function        ' num2str(
       green_func)];
61 C{25,1} = ['      number_terms_taylor_expansion      ' num2str(
       taylor_expans)];
62 C{26,1} = ['      begin "subtype"'];
63
64 %%3. Check if its a waveguide port
65 if isport == 0
66     C{27,1} = ['        begin "tl"'];
67     C{28,1} = ['        end "tl" '];
68     C{29,1} = ['      end "subtype"'];
69     C{30,1} = ['      begin "emfield"'];
70     C{31,1} = ['        char_length_emfield              1.0'];
71     C{32,1} = ['      end "emfield"'];
72     C{33,1} = ['    end "rectangular"'];
73 else            %We must write all code from here because we
       dont know exactly the col number
74     C{27,1} = ['        begin "ioport"'];
75     C{28,1} = ['          ioport             ' num2str(isport)];
76     C{29,1} = ['          mode              1'];
77     C{30,1} = ['        end "ioport"'];
78     C{31,1} = ['      end "subtype"'];
79     C{32,1} = ['      begin "emfield"'];
80     C{33,1} = ['        char_length_emfield              1.0'];
81     C{34,1} = ['      end "emfield"'];
82     C{35,1} = ['    end "rectangular"'];
83 end
84 %%4. Write RIDGE code into "file_name" file
85 %Write code. If file_name already exists its written in the
       final line
86 fprintf(fileID,'%s \n%s \n%s \n',C{:,1});
87 fclose(fileID);
88
89 end
```

Código A.4: Función "writeRectangular"

A.5. Función *writeSteps*

```
function writeSteps(file_name, waveguides_number,GUI_first_pos)
%Example of use ------>  writeSteps('filtro', 3, 220)
%Input arguments:
%   - file_name:                name of the destination file
%   - waveguides_number:        total number of waveguides in
   the design
%   - GUI_first_pos:            X axis of the block in FEST3D
   GUI

%%Index:
%1. Check if file_name file exists
                                                    (DONE)
%2. Close "waveguide" block  and start steps block
                                  (DONE)
%3. Write STEPS code into "file_name" file
                                          (DONE)

%%1. Check if file_name file exists
%filename    = ['.\out\' file_name '.txt'];
filename    = [file_name '.fest3'];
fileID      = fopen(filename,'a+');

%%2. Close "waveguide" block  and start steps block
C{1,1}          = ['  end "waveguides"'];
C{end+1,1}      = ['  begin "discontinuities"'];
for i = 1:waveguides_number-1
    GUI_x_pos  = GUI_first_pos + (i-1)*40;
    connect_from    = i;
    connect_to      = i+1;
    C{end+1,1} = ['    begin "step"'];
    C{end+1,1} = ['      number                ' num2str(i)];
    C{end+1,1} = ['      begin "gui"'];
    C{end+1,1} = ['        x_pos               ' num2str(
       GUI_x_pos)];
    C{end+1,1} = ['        y_pos               180'];
    C{end+1,1} = ['      end "gui"'];
    C{end+1,1} = ['      begin "multipactor"'];
    C{end+1,1} = ['        analize                          "
       false"'];
    C{end+1,1} = ['        use_general_settings             "
       true"'];
    C{end+1,1} = ['      end "multipactor"'];
    C{end+1,1} = ['      begin "corona"'];
    C{end+1,1} = ['        analize                          "
       false"'];
```

```
37      C{end+1,1} = ['        use_general_settings                "
            true"'];
38      C{end+1,1} = ['      end "corona"'];
39      C{end+1,1} = ['      begin "ports"'];
40      C{end+1,1} = ['        begin "port"'];
41      C{end+1,1} = ['          waveguide          ' num2str(
            connect_from)];
42      C{end+1,1} = ['        end "port"'];
43      C{end+1,1} = ['        begin "port_disc"'];
44      C{end+1,1} = ['          waveguide          ' num2str(
            connect_to)];
45      C{end+1,1} = ['          x_offset           0.0'];
46      C{end+1,1} = ['          y_offset           0.0'];
47      C{end+1,1} = ['          phi                0.0'];
48      C{end+1,1} = ['        end "port_disc"'];
49      C{end+1,1} = ['      end "ports"'];
50      C{end+1,1} = ['      begin "emfield"'];
51      C{end+1,1} = ['        char_length_emfield                1.0'
            ];
52      C{end+1,1} = ['      end "emfield"'];
53      C{end+1,1} = ['    end "step"'];
54  end
55  C{end+1,1}     = ['  end "discontinuities"'];
56  C{end+1,1}     = ['end "fest-2020"'];
57
58  %%3. Write STEPS code into "file_name" file
59  %Write code. If file_name already exists its written in the
        final line
60  fprintf(fileID,'%s \n%s \n%s \n',C{:,1});
61  fclose(fileID);
62  end
```

Código A.5: Función "writeSteps"