UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CARRERA DE CIENCIA DE LA COMPUTACIÓN

Diseño de dispositivos nanofotónicos resilientes a errores de fabricación usando algoritmos de optimización

TESIS

Para optar el título profesional de Licenciado en Ciencia de la Computación

AUTOR:

José Leonidas García Gonzales

ASESOR

Jorge Luis Gonzalez Reaño

Lima - Perú

12 de junio de 2022

Índice general

Pág.

Índice de tablas

Índice de figuras

Capítulo 1

Motivación y Contexto

1.1 Introducción

La fotónica es la ciencia que estudia la generación, detección y manipulación de la luz. Los principales beneficios que ofrece son (Shen *et al.*, 2019): (i) elevado ancho de banda en comunicaciones, (ii) bajo consumo energético, (iii) interconexiones ópticas independientes de la distancia. Actualmente, existen diversas propuestas y aplicaciones que aprovechan estos beneficio, por ejemplo: (i) interconexiones ópticas en centrales de datos (Shen *et al.*, 2019), Agregar +1 paper (ii) redes neuronales ópticas (Shen *et al.*, 2017) e Agregar +1 paper (iii) internet de las cosas (Li *et al.*, 2021). Agregar +1 paper

La integración en sistemas de cómputo Un problema presentado en el Top500 sistemas de computación de alto desempeño (HPC, por sus siglas en Inglés) es que desde el año 2010, el ratio entre el ancho de banda entre nodos y el poder de procesamiento por nodo ($byte/FLOP$) ha decrecido en un factor de seis. Es decir, se está llegando a un punto donde la capacidad para interconectar nodos está limitando el desempeño de sistemas HPC en programas que hacen uso extensivo de transferencias de memoria. Ante este problema, los avances en la fotónica en silicio (SiP) integrada se presenta como una de las principales alternativas de solución. Esta propuesta puede realizar interconexiones a distancias del orden de metros, manteniendo un elevado ancho de banda y bajo consumo energético (Anderson *et al.*, 2018, Shen *et al.*, 2019).

A pesar del avance en el diseño e integración de dispositivos SiP, en términos de cantidad de dispositivos por chip, la fotónica integrada aún se encuentra en la misma etapa de expansión que tenía la electrónica en los años 1970s (Lukas Chrostowski, 2010). Sin

embargo, ya existen procesos de fabricación estándar en *foundries* para fabricar chips SiP, a un precio accesible, utilizando procesos CMOS a través del *process design kit* (PDK) (Bogaerts y Chrostowski, 2018). De este modo, se facilita el desarrollo de esta tecnología.

En el área de SiP integrada, la alta densidad de fabricación es un desafío porque se requiere mantener eficiencia en el chip a nivel de sistema fotónico; por ello, se está buscando optimizar dispositivos fundamentales que lo compongan (Vuckovic, 2019). Para esto existen dos estrategias principales: (i) diseño tradicional (Huang y Ouyang, 2018, Hughes y Fan, 2016, Song y Xie, 2008) y (ii) diseño inverso (Gregory *et al.*, 2015, Malheiros-Silveira y Delalibera, 2020, Su *et al.*, 2020).

La figura 1.1 presenta una comparación de los dos tipos de diseño (tradicional e inverso) en dos dispositivos: (i) *bend-90°* y (ii) *wavelength demultiplexer* de dos canales (WDM). Como se observa en la figura 1.1, en el diseño tradicional (izquierda) se define el dispositivo con geometrías simples que permiten obtener funciones analíticas de sus propiedades físicas (Hughes y Fan, 2016, Song y Xie, 2008). Esto se realiza para poder optimizar la función obtenida a partir de los parámetros que la definan. Dicha optimización se suele ejecutar haciendo un barrido de los parámetros, con algoritmos genéticos o usando *particle swarm optimization*. Esta técnica es un enfoque simple, pero que ha obtenido buenos resultados (Su *et al.*, 2020).

Sin embargo, existen tres grandes inconvenientes con el diseño tradicional. Primero, solamente se explora una pequeña fracción de todos los posibles diseños. Segundo, por lo general no es conocido el límite de rendimiento del dispositivo (Molesky *et al.*, 2018). Tercero, al trabajar en la escala de nanómetros, existen casos como el *bend-90°* y WDM que presentan un bajo rendimiento (Su *et al.*, 2020).

Por el otro lado, en el diseño inverso, mostrado en la Figura 1.1 (derecha), las geometrías resultantes no están limitadas a diseños intuitivos o regulares. Esta técnica busca hacer una mayor exploración de todos los posibles diseños. Para ello, se define geometrías arbitrarias y se usa simulaciones computacionales para determinar las propiedades

(a) *Bend* con diseño tradicional.

(b) *Bend* obtenido con diseño inverso. Extraído de (Su *et al.*, 2020).

(c) WDM con diseño tradicional.

(d) WDM obtenido con diseño inverso. Extraído de (Su *et al.*, 2020).

FIGURA 1.1: Diseños tradicionales y obtenidos a partir de diseño inverso de un *bend* y un WDM.

físicas del dispositivo, propiedades que son usadas para formular una función objetivo. Finalmente, se aplican algoritmos de optimización para encontrar valores óptimos de esta función (Molesky *et al.*, 2018, Su *et al.*, 2020). Para una descripción detallada de los distintos algoritmos de optimización comúnmente empleados, por favor revisar Campbell *et al.* (2019), Elsawy *et al.* (2020), Schneider *et al.* (2019).

El diseño inverso ha logrado conseguir mejores resultados que los obtenidos por el diseño tradicional por lo que ha ganado interés en el área de fotónica durante los últimos

20 años (Campbell *et al.*, 2019, Molesky *et al.*, 2018, Su *et al.*, 2018). Sin embargo, existen cuatro grandes desafíos con este planteamiento: (i) el espacio de búsqueda es exponencial (Vuckovic, 2019), (ii) las simulaciones computacionales son costosas en términos de tiempo y memoria (Kudyshev *et al.*, 2020), (iii) el espacio de búsqueda es no convexo (Su *et al.*, 2018) y (iv) no todos los diseños son fabricables (Su *et al.*, 2020).

Y aún cuando una cantidad considerable de investigaciones estudian dispositivos SiP (Malheiros-Silveira y Delalibera, 2020, Su *et al.*, 2018), sigue siendo necesario estudiar dispositivos SiP fundamentales para conseguir chips de SiP integrada densos y eficientes. Los avances en esta área benefiarán indirectamente al desarrollo de Ciencia de la Computación con mejoras en el *hardware* utilizado para ejecutar eficientemente, por ejemplo, programas de inteligencia artificial o de computación de alto desempeño.

De este modo, el presente trabajo se centra en estudiar dos dispositivos SiP fundamentales: (i) *bend-90°* y (ii) *wavelength demultiplexer* de dos canales. De aquí en adelante nos referiremos a estos simplemente como *bend* y WDM. Por lo tanto, el objetivo de esta tesis es aplicar el conocimiento en computación para encontrar diseños de estos dispositivos con eficiencias mayores al 90 % y resilientes a errores de fabricación, trabajando en la escala de nanómetros. Con el fin de solucionar este problema se empleará diseño inverso para encontrar geometrías con las eficiencias deseadas, para ello se realizará simulaciones computacionales con cinco algoritmos de optimización populares. De este modo, se espera obtener diseños cuyas simulaciones presenten las características señaladas y que, potencialmente, muestren propiedades similares al fabricarse.

El presente documento está organizado de la siguiente manera:

El capítulo 1 brinda una introducción al tema de investigación, describe el problema a detalle, justifica la relevancia de resolverlo, define los objetivos y señala los aportes del trabajo.

El capítulo 2 desarrolla conceptos fundamentales en fotónica necesarios para entender el resto del documento.

El capítulo 3 presenta una revisión del estado del arte en diseño inverso para optimizar un *bend* y WDM.

El capítulo 4 desbribe la metodología usada en la investigación.

Por último, el capítulo 5 muestra los resultados obtenidos con los distintos casos de estudio.

1.2 Descripción del Problema

Para poder describir el funcionamiento de un dispositivo se calcula la distribución del campo eléctrico, para ello se resuelven las ecuaciones de Maxwell (Schneider *et al.*, 2019). Una forma de encontrar la solución a estas ecuaciones en cualquier geometría es utilizando un método numérico llamado diferencias finitas en el dominio de frecuencias (FDFD) (Su *et al.*, 2020). Con este planteamiento se selecciona una región rectangular a optimizar y se la divide en $n \times m$ píxeles como si fuera una imagen, ver Figura 1.2. Luego, cada píxel se rellena con dos posibles materiales: óxido de silicio (SiO_2) o silicio (Si) (Molesky *et al.*, 2018).

El diseño inverso comienza definiendo los requerimientos del dispositivo para luego tratar de buscar entre los $2^{n \times m}$ posibles diseños algún candidato que se adapte a lo que se busca (Molesky *et al.*, 2018, Su *et al.*, 2020). Como prueba de concepto, trabajos como el de Malheiros-Silveira y Delalibera (2020) parametrizaron $2^{10 \times 10}$ posibles geometrías. Así, se presentan algunas dificultades con esta estrategia:

1. No es viable evaluar todos los posibles diseños por haber un número excesivamente elevado de ellos (Vuckovic, 2019).

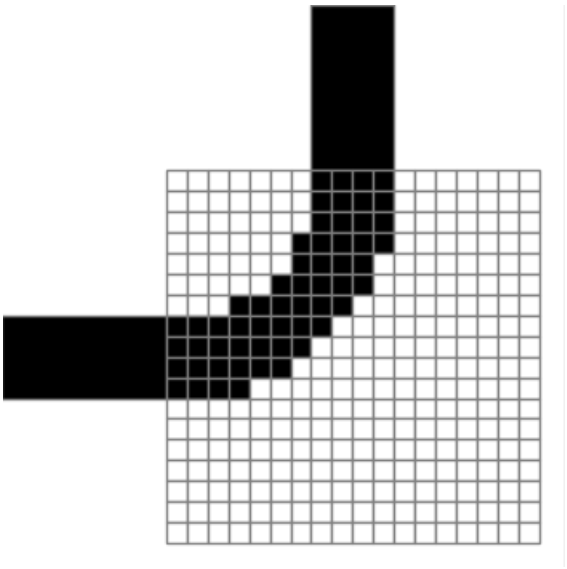


FIGURA 1.2: *Bend* con una región de diseño discretizada en 18×18 píxeles. Cada píxel negro representa la presencia de Si y cada píxel blanco de SiO_2.

2. Las simulaciones computacionales son muy costosas en términos de memoria y tiempo (Kudyshev *et al.*, 2020).

3. El espacio de búsqueda es no convexo (Su *et al.*, 2018).

4. No todos los diseños son fabricables por limitaciones físicas (Su *et al.*, 2020).

5. Cada dispositivo es una clase distinta de problema, es decir, no necesariamente funcionará la misma estrategia para cada dispositivo (Molesky *et al.*, 2018).

Además, la fabricación viene con otros desafíos, principalmente:

1. Errores de precisión (Piggott *et al.*, 2017).

2. Sensibilidad ante cambios de temperatura (Vuckovic, 2019).

Considerando las anteriores dificultades, el problema es usar diseño inverso y encontrar geometrías que muestren buen desempeño en simulaciones computacionales y que

puedan asegurar mantener un óptimo funcionamiento al ser fabricados. Este problema se estudiará para dos dispositivos nanofotónicos (i) *bend* y (ii) WDM.

1.3 Justificación

El *bend* y WDM son dispositivos SiP fundamentales que tienen aplicación directa, por ejemplo, en sistemas HPC (Shen *et al.*, 2017). Así, las mejoras de estos ayudará indirectamente al desarrollo de Ciencia de la Computación brindando, potencialmente, un mejor *hardware* para programas de inteligencia artificial, HPC, entre otros. Por otro lado, desde el punto de vista computacional, este problema es interesante porque ya hay estrategias computacionales conocidas para resolverlo, desde algoritmos evolutivos (Hansen, 2016) hasta redes neuronales (Goodfellow *et al.*, 2015) y *depth learning* (Malkiel *et al.*, 2018). Además, debido al alto costo computacional de las simulaciones (Schneider *et al.*, 2019), el trabajo requiere de computación de alto desempeño. Así, es probable que se pueda obtener buenos resultados en la investigación aplicando el conocimiento ya existente en computación.

1.4 Objetivos

- Diseñar un *bend* y WDM con eficiencias mayores al 90 % y resiliente a errores de fabricación.
    - Seleccionar una estrategia de parametrización que asegure facilidad de fabricación.
    - Definir una función objetivo que encapsule las propiedades buscadas en cada dispositivo.
    - Encontrar geometrías con valores óptimos de la función objetivo en simulaciones computacionales.

- Encontrar geometrías resilientes a posibles errores de fabricación de dilatación o contracción.

- Comparar el desempeño y la convergencia de cinco algoritmos de optimización populares usados para optimizar dispositivos nanofotónicos.

  - Implementar el proceso de optimización de los dispositivos asegurando aprovechar los recursos GPU de un *cluster*.

Capítulo 2

Marco Teórico

En el presente capítulo se introducen conceptos fundamentales relacionados al diseño inverso de dispositivos fotónicos. Para ello se desarrolla seis secciones. Primero, se describen las propiedades físicas de interés de un *bend* y WDM. Segundo, se explica como parametrizar la región de diseño de estos dispositivos. Tercero, se detalla los pasos necesarios para poder simular computacionalmente los diseños realizados. Cuarto, se explica la estrategia de optimización a seguir con la parametrización señalada. Quinto, se describen tres algoritmos que se usaran en el presente trabajo: (i) algoritmos genéticos (GA), (ii) *particle swarm optimization* (CMA), (iii) *covariance matrix adaptation evolution strategy* (CMA-ES). Finalmente, se expone que transformaciones se puede aplicar dentro de la estrategia a seguir.

TODO: Actualizar esto al final.

2.1 Dispositivos de estudio

2.1.1 *Bend*

Un *bend* es un dispositivo fotónico que se encarga de guiar el giro de un haz de ondas.

En general, al estudiar dispositivos fotónicos es de especial interés la distribución del campo eléctrico ($\boldsymbol{E}$). Este nos permite visualizar la distribución de la energía en un dispositivo calculando lo siguiente:

$$|\boldsymbol{E}|^2 = |\boldsymbol{E_x}|^2 + |\boldsymbol{E_y}|^2 + |\boldsymbol{E_z}|^2, \tag{2.1}$$

donde $\boldsymbol{E_x}, \boldsymbol{E_y}, \boldsymbol{E_z}$ representan las componentes del campo eléctrico en los ejes x, y, z, respectivamente (Lukas Chrostowski, 2010).

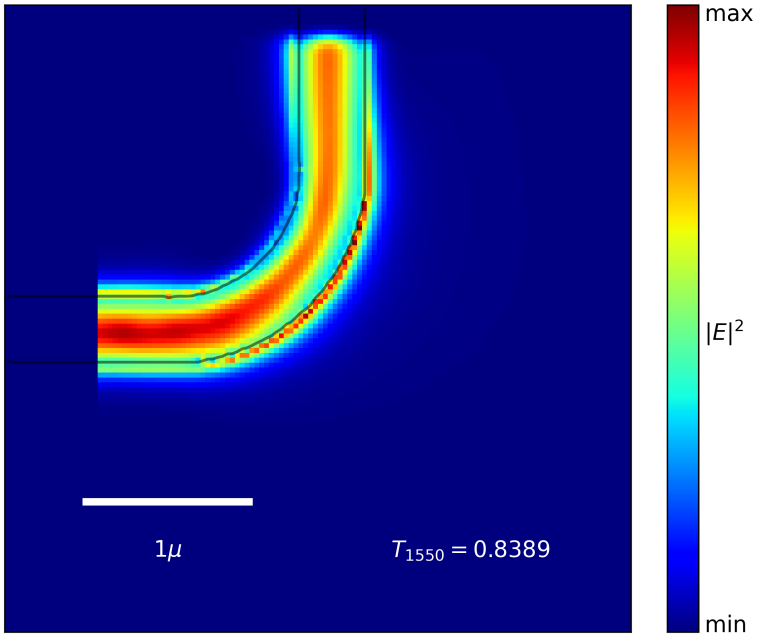


FIGURA 2.1: Representación de $|\boldsymbol{E}|^2$ de un *bend* de $1\mu m$ de radio obtenido con una simulación en 3D FDFD con SPINS bajo una resolución de $30nm$.

Tradicionalmente, un *bend* consiste en una una guía de onda horizontal usada como entrada y una guía de onda vertical usada como salida, estas son conectadas por una guía de onda con la forma de un cuarto de circunferencia de radio r y con el mismo grosor de las guías de onda de entrada y salida. En la Figura 2.1 se muestra un *bend* tradicional de radio $r = 1\mu m$. Como se observa en la imagen, parte significativa de la energía se pierde en la región curva. Esto se debe a que el radio de curvatura es muy pequeño, con un valor más grande (e.g. $r = 10\mu m$) las pérdidas se vuelven casi nulas (Lukas Chrostowski, 2010).

Para evitar ambigüedades, dos puntos adicionales a remarcar son: (i) cuando nos referimos al radio estamos haciendo mención al radio medio de curvatura y (ii) todos los diseños mostrados en este trabajo tienen un profundidad de 220nm.

Observar en una gráfica el valor de $|\boldsymbol{E}|^2$ nos ayuda a entender el funcionamiento de un dispositivo. Por otro lado, una manera de cuantificar que tan bien funciona un diseño es mediante el cálculo de la transmitancia (T). Este valor se define como la relación entre la potencia del flujo que sale del dispositivo con la potencia del flujo que ingresa (Christiansen y Sigmund, 2021b).

Seguidamente, sea λ la longitud de onda de la entrada y $\boldsymbol{P}$ los parámetros que caracterizan al diseño, denotaremos como $T_\lambda(\boldsymbol{P})$ a la transmitancia asociada al dispositivo obtenido con la parametrización $\boldsymbol{P}$ en la longitud de onda λ. Luego, definimos la función objetivo (f_{obj}) para un *bend*, también conocido en el área como figura de mérito (FOM), mediante la siguiente ecuación (Su *et al.*, 2020):

$$f_{obj}(\boldsymbol{P}) = max\left\{T_{1550}(\boldsymbol{P})\right\}. \tag{2.2}$$

En síntesis, la idea detrás de estas definiciones es describir un *bend* mediante una parametrización $\boldsymbol{P}$ (Sección 2.2). Luego, usando algoritmos de optimización, buscar entre las distintas combinaciones de los parámetros aquella configuración que optimice la función f_{obj} (Sección 2.5). De este modo, estaremos encontrando un diseño con una elevada transmitancia, es decir con un buen desempeño.

2.1.2 *Wavelength Demultiplexer* (WDM)

Un WDM es un dispositivo fotónico que se encarga de guiar un haz de ondas de acuerdo a su longitud de onda. Por ejemplo, estos pueden trabajar con dos longitudes de

onda y guían las de una longitud por la guía de onda superior y las de otra longitud por la guía de onda inferior.

Análogo al caso del *bend*, utilizaremos la transmitancia para cuantificar el desempeño del dispositivo. Pero, en el presente trabajo usaremos un WDM con dos guías de salida, por ello, utilizaremos como notación $T_{\lambda}^{(1)}(\boldsymbol{P})$ para representar la transmitancia en la guía de salida superior cuando se recibe un haz de longitud de onda λ en un diseño descrito por los parámetros $\boldsymbol{P}$ y $T_{\lambda}^{(2)}(\boldsymbol{P})$ para la guía de salida inferior.

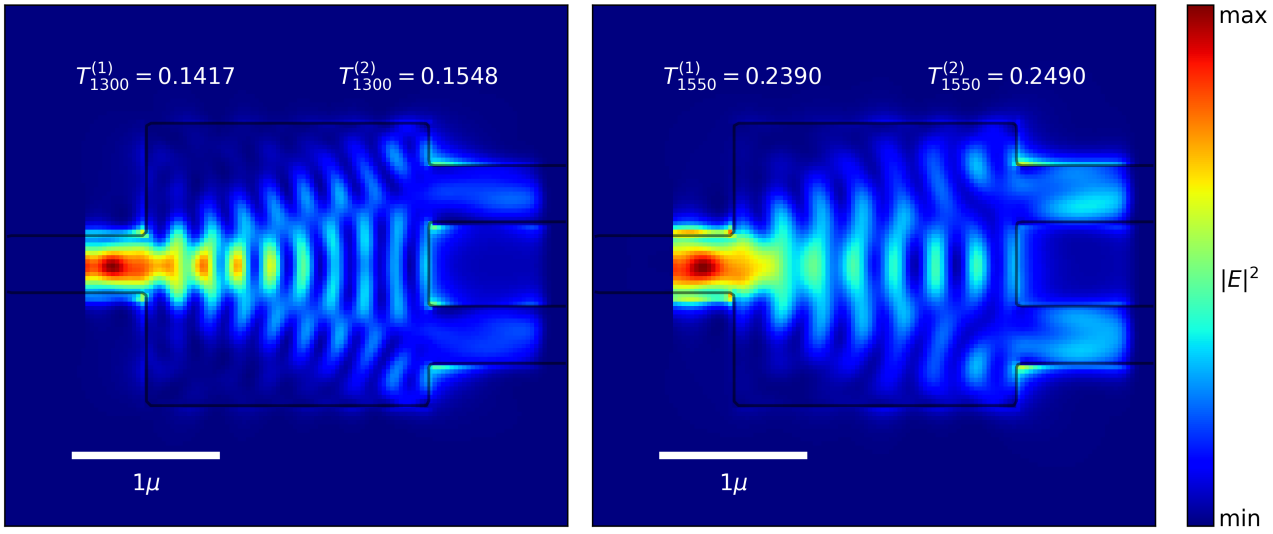


FIGURA 2.2: Representación de $|\boldsymbol{E}|^2$ de un WDM obtenido con una simulación en 3D FDFD con SPINS bajo una resolución de $30nm$.

En la Figura 2.2 se muestra un WDM donde las guías de onda son unidas por una región rectangular de $2.0\mu m \times 2.0\mu m$. En el lado izquierdo se muestra la representación de $|\boldsymbol{E}|^2$ cuando se usa como entrada un haz de ondas de $1300nm$ de longitud, en el lado derecho la entrada es de $1550nm$. En ambos casos el diseño está funcionando más como un *splitter* que como un WDM. Un *splitter* es un dispositivo fotónico que divide la potencia del flujo de la entrada por las guías de salida en una determinada proporción. El diseño intuitivo para este dispositivo es el presentado en la imagen. Similar al caso del *bend*, con unas dimensiones un poco más grandes se puede conseguir pérdidas casi nulas

de energía (Lukas Chrostowski, 2010). Sin embargo, como se observa en los valores de la transmitancia, este diseño intuitivo no es apropiado para un WDM.

Basándonos en Su *et al.* (2020), definimos su FOM como

$$f_{obj}(\boldsymbol{P}) = max \left\{ \frac{\left(T^{(1)}_{1300}(\boldsymbol{P})\right)^2 + \left(1 - T^{(2)}_{1300}(\boldsymbol{P})\right)^2 + \left(1 - T^{(1)}_{1550}(\boldsymbol{P})\right)^2 + \left(T^{(2)}_{1550}(\boldsymbol{P})\right)^2}{4} \right\}. \tag{2.3}$$

La Ecuación 2.3 busca maximizar la transmitancia por la guía de onda superior y minimizarla para la guía de onda inferior cuando se recibe una longitud de onda de $1300nm$ y lo contrario para una longitud de onda de $1550nm$. Cabe destacar que la división por cuatro se realiza para asegurar que f_{obj} solamente tenga valores en el intervalo $[0, 1]$, al igual que sucede con la función objetivo del *bend*.

La idea para optimizar un WDM es la misma descrita en la anterior sección. En el resto del capítulo se describe en más detalle los otros pasos necesarios para lograr esto.

2.2 Parametrización

Tanto para el *bend* como para el WDM se define una región de diseño mediante una parametrización ($\boldsymbol{P}$) que pueda mapear un gran conjunto de dispositivos. Una de la estrategias más populares para esta tarea es usar parametrización basada en píxeles, esta consiste en definir $\boldsymbol{P}$ como una matriz con valores en el intervalo $[0, 1]$ (Molesky *et al.*, 2018).

Por ejemplo, en la Figura 2.3 se ha definido la matriz $\boldsymbol{P} : [1, n] \times [1, m] \to [0, 1]$ con $n = m = 18$ y se ha graficado sus valores usando escala gris (0 corresponde al color blanco, 1 al negro y los demás valores a diferentes intensidades del gris). De esta manera

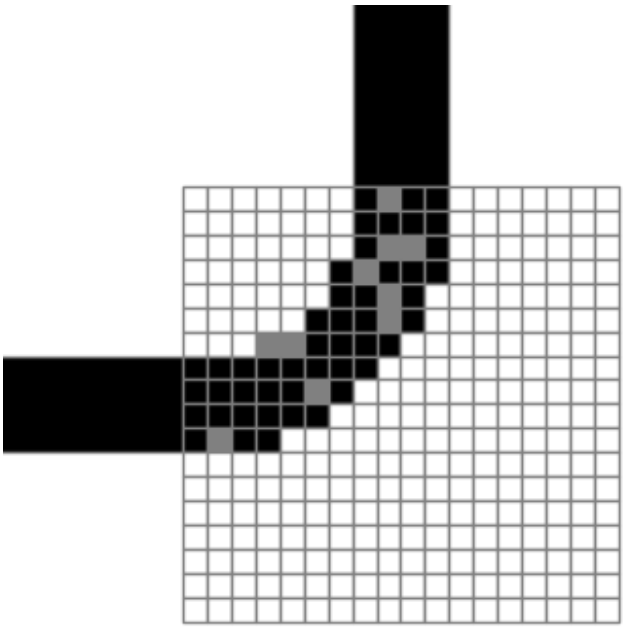


FIGURA 2.3: Parametrización basada en píxeles para un *bend* definiendo $\boldsymbol{P}$ como una matriz de 18×18.

observamos como $\boldsymbol{P}$ logra definir la geometría de un diseño. Adicionalmente, conforme se incrementa el valor de n, m se logra definir detalles con mayor precisión.

Sin embargo, lo anterior solo nos permite describir la geometría de los diseños, no sus propiedades. Por ello, es necesario asociar a $\boldsymbol{P}$ alguna propiedad física. Esto lo realizamos calculando la permitividad (ε) mediante la siguiente ecuación:

$$\varepsilon(x, y) = \varepsilon_{Si} + (1 - p_{x,y})\varepsilon_{SiO_2} \mid x \in [1, n] \wedge y \in [1, m] \wedge p_{x,y} \in \boldsymbol{P}, \qquad (2.4)$$

donde $\varepsilon_{Si} = 3.48$ es la permitividad del silicio (Si), $\varepsilon_{SiO_2} = 1.44$ es la permitividad del óxido de silicio ($SiO2$) y $p_{x,y}$ es el elemento de $\boldsymbol{P}$ ubicado en la fila x, columna y.

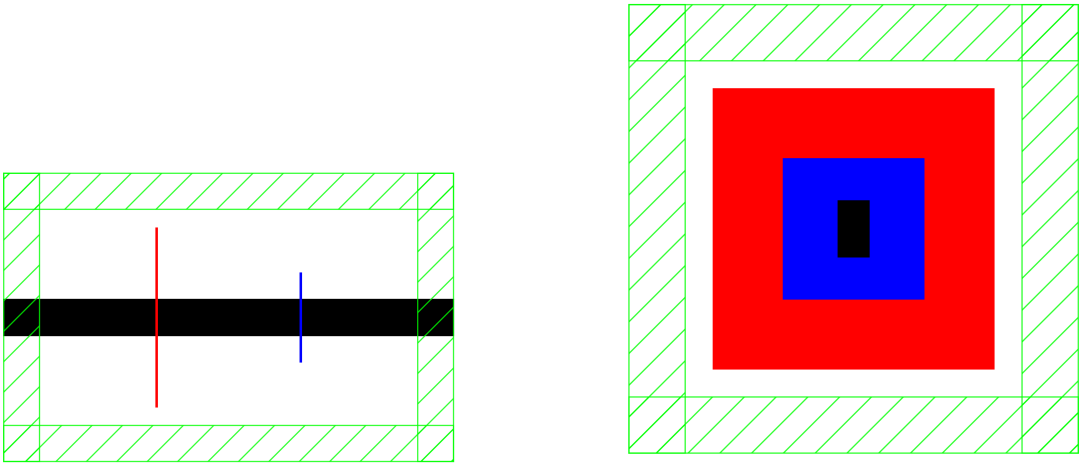
Con la Ecuación 2.4 estamos asociando a cada rectángulo de la geometría descrita por $\boldsymbol{P}$ un valor de permitividad en el rango $[\varepsilon_{SiO2} = 1.44, 3.48 = \varepsilon_{Si}]$. De esta manera, $p_{x,y} = 1$ describe que el rectángulo ubicado en la posición (x, y) es de silicio, un valor de $p_{x,y} = 0$ describe la presencia de óxido de silicio y $0 < p_{x,y} < 1$ hace referencia a algún material cuya permitividad es $\varepsilon(x, y)$.

No obstante, un inconveniente de lo descrito es que $\boldsymbol{P}$ puede mapear a materiales inexistestes (regiones grises), en la Sección 2.4 se estudia en más detalle esta dificultad. Por otro lado, con la descripción de la permitividad ya podemos calcular los campos eléctricos con lo cual se puede obtener el valor de f_{obj} definido para el *bend* y WDM. Por tal motivo, se realiza una simulación electromagnética y se resuelve las inversas de las ecuaciones de Maxwell, esto permite obtener los campos eléctricos (Su *et al.*, 2020). Afortunadamente, para este proceso se puede utilizar distintos programas que utilizan diversos métodos numéricos para realizar los cálculos (Sección 2.3).

2.3 Simulación

Una vez tenemos definido un dispositivo con regiones fijas (guías de onda) y una región de diseño (descrita por la parametrización), es necesario incorporar tres elementos adicionales (Oskooi *et al.*, 2010, Su *et al.*, 2020):

- **Fuente:** Suele representarse como un rectángulo en un plano perpendicular al flujo que pasa por el lugar donde este se ubica. Simula la emición de un haz de ondas por el diseño.
- **Monitores:** Suelen representarse como un rectángulo similar a la fuente. Capturan información en su ubicación (e.g. valores del campo eléctrico).
- **PML:** Representan las condiciones de frontera en la simulación. Se utilizan para limitar el espacio donde se deberá realizar las simulaciones computacionales.



(a) Vista frontal (b) Vista lateral

FIGURA 2.4: Configuración de simulación para una guía de onda.

En la Figura 2.4 se ilustra una guía de onda con los elementos mencionados. El rectángulo de color negro representa la guía de onda, la región roja la fuente, la región azul el monitor y lo verde el PML. Guiándonos de la figura de la izquierda, al realizar una simulación con esta configuración el flujo iría de la región roja a la región azul. Sin embargo, este seguiría expandiéndose por la guía de onda hasta llegar a la región verde (PML). En palabras sencillas, podemos interpretar el PML como una caja que permite aislar nuestro sistema. Por otro lado, la figura de la derecha representa el mismo diseño pero desde una vista lateral (recordemos que son diseños en 3D). Aquí podemos evidenciar como la fuente y el monitor son representados por rectángulos, no por rectas.

Siguiendo lo descrito, para configurar y ejecutar simulaciones de un *bend* y WDM podemos utilizar diversos programas. Dos librerías de Python de código abierto que permiten hacer estas tareas son MEEP (Oskooi *et al.*, 2010) y SPINS (Su *et al.*, 2020). Una evaluación cualitativa de sus funcionalidades puede ser observada en la Tabla 2.1.

Como se describe en Tabla 2.1, desde mi punto de vista, MEEP es un programa eficiente y con más funcionalidades que SPINS. Sin embargo, estas ventajas se reflejan

Librería	MEEP	SPINS
Usabilidad	Dificil	Moderada
Eficiencia	Alta	Moderada-Alta
Funcionalidad	Extensa	Básica-necesaria
¿Soporta GPU?	No	Sí
¿Implementa el método *adjoint*?	Sí	Sí

TABLA 2.1: Evaluación cualitativa de las librerías MEEP y SPINS.

en una usabilidad más complicada. Además, en realidad, la eficiencia de SPINS es muy destacable en diversas situaciones, en parte gracias a su implementación en GPU para las simulaciones. Finalmente, sobre el método *adjoint* no hay inconvenientes en ninguno de los programas. Este método permite calcular eficientemente (independientemente de la dimensión de $\boldsymbol{P}$) el valor de $\frac{\mathrm{d}f_{obj}(\boldsymbol{P})}{\mathrm{d}\boldsymbol{P}}$ (Molesky *et al.*, 2018).

Por otro lado, ambas librerías utilizan métodos numéricos y permiten simulaciones en 1D, 2D y 3D. Particularmente, podemos definir un parámetro dx para discretizar nuestro espacio de simulación en una malla de cuadrados de dimensiones $dx \times dx$. En general, cuanto más pequeño sea este valor los resultados serán más precisos, pero la cantidad de memoría y el tiempo de simulación se incrementarán considerablemente. Es importante señalar que debemos procurar trabajar con un valor de dx menor a las dimensiones de nuestros rectángulos de la parametrización para poder simular nuestras geometrías con mejor precisión.

Adicionalmente, mientras MEEP utiliza un método conocido como FDTD para las simulaciones, SPINS usa FDFD. Por un lado, MEEP permite calcular la función objetivo en un rango de longitudes de onda usando una sola simulación. En cambio, para realizar esta tarea SPINS debe realizar una simulación por cada longitud de onda.

2.4 Estrategia de optimización

Como fue descrito en la Sección 2.2, podemos tener diseños que no puedan ser fabricados. Para evitar esto se necesita obtener que $p_{x,y}$ de la Ecuación 2.4 tenga valores enteros. Sin embargo, al optimizar la función f_{obj} no podemos asegurar esta condición. Ante esta dificultad, Su *et al.* (2020) trabaja en dos etapas:

1. **Optimización continua**

   En esta etapa se optimiza la función f_{obj} sin imponer ninguna restricción. En la Sección 2.5 se detalla algoritmos que suelen usarse para este fin.

2. **Optimización discreta**

   Se utiliza el resultado de la optimización continua como punto inicial para el algoritmo de optimización que se escoja. Luego, se trabaja por iteraciones. En cada iteración se aplica una transformación a cada diseño antes de evaluar su FOM. Esta transformación se escoge de tal manera que ayude a ir convergiendo a un diseño fabricable, es decir, a tener $p_{x,y} = 0$ o $p_{x,y} = 1$, más detalles en la Sección 2.6. Así, la idea de realizarlo por iteraciones es ir discretizando nuestro diseño de forma suave e intentando mantener un buen valor del FOM. Por ello, es crucial utilizar el resultado de una iteración como punto inicial de la próxima.

2.5 Algoritmos de Optimización

Como observamos en la anterior sección, es necesario escoger algún algoritmo que nos permita optimizar la función f_{obj}. Por este motivo, se presentan los siguientes tres algoritmos que serán usados en el presente trabajo.

2.5.1 *Genetic Algorithms* (GA)

TODO: Actualizar nomenclatura.

Como se describe en el Algoritmo 1 (Kochenderfer y Wheeler, 2019), la idea es comenzar generando una población (*population*) de n individuos representados por p parámetros, línea 1. Los siguientes tres pasos se ejecutan por k iteraciones. Primero, se realiza un proceso de selección para obtener los mejores individuos (*parents*), línea 3. Segundo, los seleccionados se encargan de producir la nueva generación (*children*), línea 4. Tercero, la nueva generación muta obteniendo nuevas características, línea 5.

Algorithm 1: Estructura de un algoritmo genético

```
1 population = generate_population(n, p)
2 for t = 0; t < k; t++ do
3     parents = select(population)
4     children = crossover(population, parents)
5     population = mutation(children)
```

Entrando en más detalle, para nuestro caso tenemos:

- $generate\_population(n, p)$: retorna n vectores de dimensión p con valores aleatorios en $U(0, 1)$.
- $select(population)$: retorna $number\_selected\_GA$ individuos de acuerdo a la probabilidad $prob_i$ dada por la ecuación

$$prob_i = \frac{f_{obj}^{(i)} - min(f_{obj})}{\sum_j (f_{obj}^{(j)} - min(f_{obj}))}, \tag{2.5}$$

  donde $f_{obj}^{(i)}$ está asociado con el $i-$ésimo individuo.
- $crossover(population, parents)$: retorna n vectores de dimensión p. El $i-$ésimo vector es la combinación de dos padres aleatorios pa_i y pb_i seleccionados de $parents$

y su $j-$ésimo parámetro es escogido con igual probabilidad entre el $j-$ésimo parámetro de $parents[pa_i]$ y $parents[pb_i]$.

- $mutation(children)$: retorna lo que recibe, pero a cada atributo de cada individuo se le agrega un valor en $U(-range\_GA, range\_GA)$ con probabilidad de 0.5.

2.5.2 *Gradient Genetic Algorithms* (G-GA)

TODO: Completar esto

2.5.3 *Particle Swarm Optimization* (PSO)

TODO: Actualizar nomenclatura.

Podemos pensar este algoritmo como un caso especial del Algoritmo 1 (Kochenderfer y Wheeler, 2019, Prosopio-Galarza *et al.*, 2019). La idea es visualizar el $i-$ésimo individuo como una partícula definida por: (i) su posición $x^{(i)}$ (el vector p-dimensional asociado al $i-$ésimo individuo), (ii) su velocidad $\nu^{(i)}$ (un número real) y (iii) la mejor posición encontrada hasta el momento.

Cada particula acumula velocidad en una dirección favorable dada por: (i) la mejor posición encontrada hasta el momento por ella y (ii) la mejor posición encontrada por la población completa. Como consecuencia, los individuos se pueden mover independientemente de perturbaciones locales. Adicionalmente, agregando caminos aleatorios las particulas incorporan comportamientos impredecibles que puede permitirles encontrar potenciales mejores direcciones.

Entrando en más detalle, siguiendo la estructura del Algoritmo 1, tenemos:

- $generate\_population(n, p)$: retorna n particulas con cada uno de sus parámetros tomando valores aleatorios en $U(0, 1)$.

- $select(population)$: retorna la particula con el mejor f_{obj}
- $crossover(population, parents)$: retorna la población luego de aplicar

$$x^{(i)} \leftarrow x^{(i)} + \nu^{(i)}, \tag{2.6}$$

$$\nu^{(i)} \leftarrow \omega\nu^{(i)} + c_1 r_1 \left(x_b^{(i)} - x^{(i)}\right) + c_2 r_2 \left(x_b - x^{(i)}\right), \tag{2.7}$$

donde x_b es la mejor posición encontrada globalmente, ω representa la tendencia de la particula de conservar su velocidad actual, c_1 y c_2 cuantifica la atracción relativa de $x_b^{(i)}$ y x_b respectivamente, y $r_1, r_2 \in U(0,1)$ representan el comportamiento impredecible.

2.5.4 *Gradient Particle Swarm Optimization* (G-PSO)

TODO: Completar esto

2.5.5 *Covariance Matrix Adapatation Evolution Strategy* (CMA-ES)

TODO: Actualizar nomenclatura.

La idea general de esta estrategia evolutiva, mostrada en el Algoritmo 2, es mantener: (i) un vector μ p-dimensional, (ii) una matriz Σ y (iii) un número σ para ir generando n inviduos p-dimensionales a partir una distribución $\mathcal{N}(\mu,\, \sigma^2\Sigma)$.

Tomar puntos de esta distribución limita el espacio de búsqueda a una hiperelipse. Luego, el algoritmo evalua puntos en esta región limitada. Usando los valores obtenidos, se puede decidir entre: (i) mover la hiperelipse a otra región del espacio de búsqueda (ii) expandir or reducir la región cubierta por la distribución. El algoritmo de CMA-ES

trabaja iterativamente sobre esta idea hasta que la hiperelipse termina casi degenerándose en un punto, potencialmente un óptimo. Para una descripción más detallada del algoritmo, revisar (Hansen, 2016, Kochenderfer y Wheeler, 2019).

Algorithm 2: CMA-ES

```
1 for t = 0; t < k; t++ do
2     sample() // Obtener n puntos de N(μ, σ²Σ)
3     update() // Ecuación 2.8
4     control() // Ecuación 2.9
5     adapt() // Ecuación 2.11
```

Entrando en más detalle del Algoritmo 2 y considerando variables globales por simplicidad, podemos resumir el procedimiento en cinco pasos. En la línea 1 simplemente se repite las siguientes líneas por k iteraciones. En la iteración t, comenzamos con la línea 2 generando n puntos p-dimensionales x_i de la distribución $\mathcal{N}(\mu, \sigma^2\Sigma)$, donde estos son ordenados descendentemente de acuerdo al valor de f_{obj}. En la línea 3 actualizamos la media μ usando una promedio ponderado dado por

$$\mu^{(t+1)} \leftarrow \sum_{i=1}^{n} w_i x_i, \tag{2.8}$$

,

donde w_i son fijos y escogidos de tal manera que proporcionen mayor contribución a los puntos con meyor f_{obj}. Esto permite mover la media μ en una dirección favorable.

Seguidamente, se necesita actualizar σ para expandir o reducir la hiperelipse en la siguiente iteración. Por este motivo, la línea 4 controla este valor mediante las ecuaciones

$$\sigma^{(t+1)} \leftarrow \sigma^{(t)} \exp\left(\frac{c_\sigma}{d_\sigma} \underbrace{\left(\frac{||p_\sigma||}{\mathbb{E}||\mathcal{N}(0, \mathbf{I})||} - 1\right)}_{\text{evolution path comparison}}\right), \tag{2.9}$$

$$\mathbb{E}||\mathcal{N}(0,\mathbf{I})|| = \sqrt{2}\left(\frac{\Gamma\left(\frac{p+1}{2}\right)}{\Gamma\left(\frac{p}{2}\right)}\right), \tag{2.10}$$

donde p_σ es una variable que acumula los pasos llevados, $c_\sigma \in [0,1]$ es una variable que determina el tiempo acumulado para p_σ y $d_\sigma \approx 1$ es un parámetro que determina el ratio de posibilidad de cambio de $\sigma^{(t+1)}$. La principal parte de la Ecuación 2.9 es el término *evolution path comparison*, aquí se compara el tamaño de p_σ con su tamaño esperado bajo selección aleatoria. De esta comparación podemos controlar si el valor de σ debe incrementarse, disminuirse o permanecer igual.

Finalmente, en la línea 5 cambiamos Σ a una dirección favorable usando

$$\Sigma^{(t+1)} \leftarrow \overbrace{\left(1 - c_1 c_c(1-h_\sigma)(2-c_c) - c_1 - c_\mu\right)\Sigma^{(t)}}^{\text{cumulative update}} + \underbrace{c_1 p_\Sigma p_\Sigma^T}_{\text{rank-one update}} + \underbrace{c_\mu \sum_{i=1}^{n} w_i' \delta^{(i)} \left(\delta^{(i)}\right)^T}_{\text{rank-}\mu\text{ update}}, \tag{2.11}$$

donde $c_\mu \leq 1$ es el radio de aprendizaje para el término *rank-μ update*, $c_1 \leq 1 - c_\mu$ es el radio de aprendizaje para el término *rank-one update*, $c_c \in [0,1]$ es el radio de aprendizaje para el término *cumulative update*, h_σ es la evaluación bajo la función unitaria usado para actualizar apropiadamente el camino evolutivo, p_Σ es un vector acumulativo usado para actualizar la matriz de covarianza, w_i' son los coeficientes de ponderación modificados y $\delta^{(i)}$ son las desviaciones seleccionadas.

En la Ecuación 2.11, el primer término (*cumulative update*) mantiene información de la anterior matriz de convarianza. El segundo término (*rank-one update*) permite expandir la distribución en una dirección favorable. El tercer término (*rank-μ update*) incrementa la búsqueda en espacios donde es probable encontrar buenas soluciones. La

combinación de estos tres términos actualiza Σ de tal manera que mueva la hiperelipse en una dirección favorable.

2.5.6 *Gradient Covariance Matrix Adapatation Evolution Strategy* (G-CMA-ES)

TODO: Completar esto

2.5.7 L-BFGS-B

TODO: Completar esto

2.5.8 MMA

TODO: Completar esto

2.6 Transformaciones

La aplicación de transformaciones a un diseño se realiza con el fin de obtener dispositivos que se puedan fabricar con mayor facilidad (Su *et al.*, 2020).

Basándonos en Zhang *et al.* (2021), una manera de asegurar que los parámetros p describan un diseño más fácil de fabricar es aplicacando la función $s(p)$ descrita como

$$\widetilde{\widetilde{\boldsymbol{P}}}(x,y) = \frac{\tanh(\beta \times \eta) + \tanh(\beta \times (p_{x,y} - \eta))}{\tanh(\beta \times \eta) + \tanh(\beta \times (1 - \eta))}, \tag{2.12}$$

donde $\eta = 0.5$ y β comienza con un valor de 1 y va incrementándose exponencialmente en cada iteración. Como se observa en la Figura 2.5, la Ecuación 2.12 se encarga

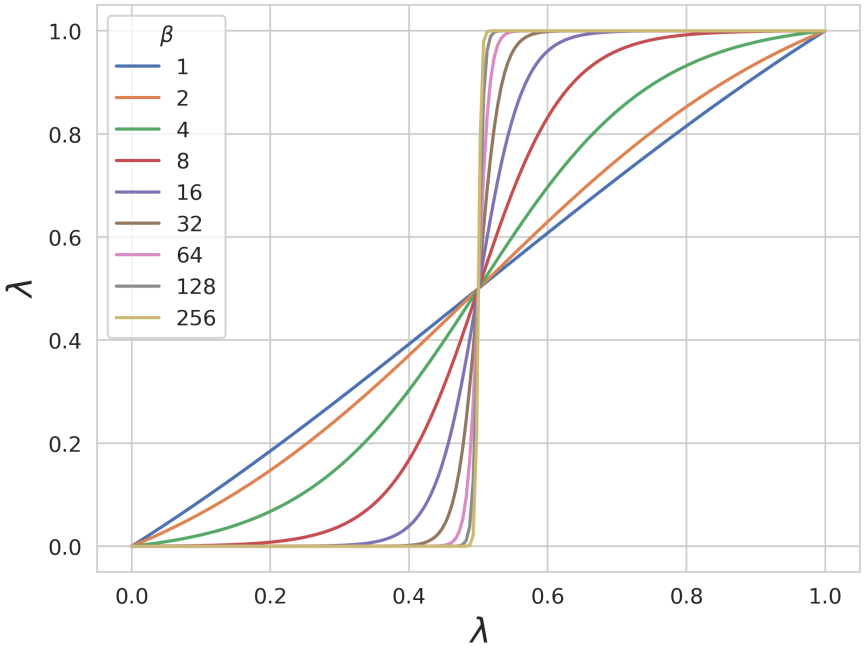


FIGURA 2.5: Función de discretización con $\eta = 0.5$ y distintos valores de β.

de ir haciendo converger los valores de la parametrización a 0 o 1 de acuerdo a cual esté más cercano. Conforme aumenta el valor de β esta convergencia es más rápida.

TODO: Agregar una descripción detallada sobre las siguientes ecuaciones.

$$\widetilde{\boldsymbol{P}}(x,y) = \frac{\sum_{(x',y')\in B_{r_f}(x,y)} w_{x,y}(x',y')A(x',y')p_{x',y'}}{\sum_{(x',y')\in B_{r_f}(x,y)} w_{x,y}(x',y')A(x',y')}, \tag{2.13}$$

$$w_{x,y}(x',y') = max(0, r_f - dis(x,y,x',y')) \tag{2.14}$$

$$dis(x,y,x',y') = |x - x'| * psize_y + |y - y'| * psize_x \tag{2.15}$$

$$\frac{\partial \widetilde{\boldsymbol{P}}(x,y)}{\partial \boldsymbol{P}(x^*,y^*)} = \frac{w_{x,y}(x^*,y^*)A(x^*,y^*)}{\sum\limits_{(x',y')\in B_{r_f}(x,y)} w_{x,y}(x',y')A(x',y')}, \tag{2.16}$$

$$\frac{\partial \widetilde{\widetilde{\boldsymbol{P}}}(x,y)}{\partial \widetilde{\boldsymbol{P}}(x,y)} = \frac{(1-\tanh^2(\beta\times(\widetilde{p}_{x,y}-\eta))\beta)}{\tanh(\beta\times\eta)+\tanh(\beta\times(1-\eta))}, \tag{2.17}$$

$$\frac{\partial f_{obj}}{\partial \boldsymbol{P}(x,y)} = \sum_{(x',y')\in B_{r_f}(x,y)} \frac{\partial f_{obj}}{\partial \widetilde{\widetilde{\boldsymbol{P}}}(x',y')} \frac{\partial \widetilde{\widetilde{\boldsymbol{P}}}(x',y')}{\partial \widetilde{\boldsymbol{P}}(x',y')} \frac{\partial \widetilde{\boldsymbol{P}}(x',y')}{\partial \boldsymbol{P}(x,y)} \tag{2.18}$$

Capítulo 3

Revisión Crítica de la Literatura

En el presente capítulo comenzaremos discutiendo sobre un trabajo que optimiza un *splitter*. Seguidamente, identificaremos dos inconvenientes con este trabajo e iremos mostrando como otras investigaciones han afrontado estos desafíos.

En Prosopio-Galarza *et al.* (2019) se optimizó un *splitter* con guías de onda fijas de altura $0.5\mu m$, donde las guías de onda de salida son separadas por $0.2\mu m$ y todas estas son unidas con una región rectangular de diseño de $2\mu m \times 1.5\mu m$. El diseño utiliza un espesor estándar de $220nm$. Con esta geometría se simulan distintos diseños dividiendo la región rectangular en ($z = 13$) segmentos uniformemente separados. Cada segmento puede variar su altura dentro del rectángulo, estos se centran de forma vertical y se van uniendo sus extremos. La representación de esta idea la podemos observar en la Figura 3.1 con $z = 5$ segmentos.

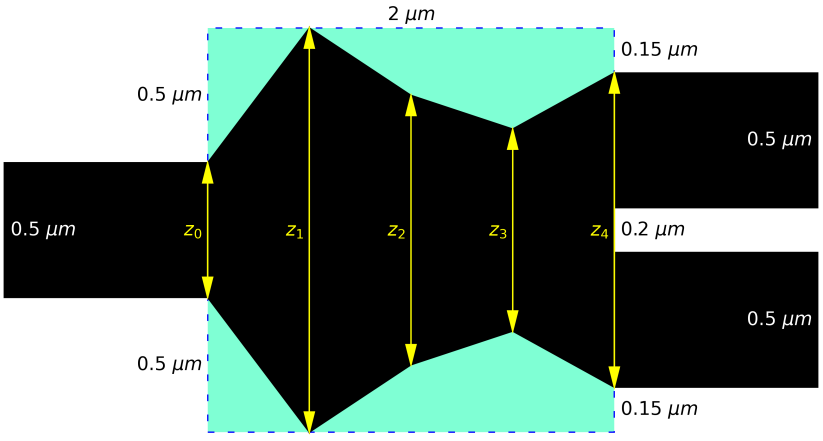


FIGURA 3.1: Diseño de un *splitter* basado en (Prosopio-Galarza *et al.*, 2019) utilizando $z = 5$ segmentos.

Como función objetivo se establece maximizar la transmitancia en la guía de onda superior trabajando con una longitud de onda de $1550nm$. Los mejores resultados son obtenidos al usar PSO como algoritmo de optimización.

Es destacable que al usar esta parametrización es posible limitar las alturas de los segmentos para asegurar obtener ángulos agudos, los cuales son los más adecuados como regla práctica de diseño (Lukas Chrostowski, 2010). Sin embargo, hay dos inconvenientes con este trabajo:

- **Inconveniente 1:** La parametrización utilizada descarta la posibilidad de diseños menos intuitivos (por ejemplo, con agujeros) que podrían ocupar menor área y mantener una buena transmitancia.
- **Inconveniente 2:** Las optimizaciones solo se repitieron una vez con apenas 30 iteraciones y una población de 14 individuos.

A continuación, vamos a desarrollar en más detalle como otros trabajos han buscado solucionar estos dos inconvenientes.

3.1 Inconveniente 1: La Parametrización

Una estrategia de parametrización interesante es la de dividir de forma uniforme la región rectangular de diseño en $n \times m$ rectángulos como si fueran píxeles. Luego, podemos considerar que un píxel negro representa la presencia de Si y uno blanco de SiO_2, como se mostró en la Figura 1.2. Aunque trabajaron con otro dispositivo, esta estrategia se aplicó en Malheiros-Silveira y Delalibera (2020) con $n = m = 10$.

Sin embargo, una estrategia aún más interesante es la asignación a cada píxel de una permitividad definida por la Ecuación 2.4, estrategia que se evaluó en Su *et al.* (2020) para optimizar un *bend* y *WDM*. El principal beneficio de esta idea es que pasamos de

tener un problema de optimización donde el resultado solo podía tomar valores enteros a uno donde podemos considerar valores reales. Con esto en consideración, se sigue la estrategia descrita en la Sección 2.4: optimización continua y optimización discreta.

Aún cuando la idea anterior es muy buena, al momento de incorporar restricciones de fabricación simplemente se considera usar parámetros más grandes que la mínima precisión que puede manejar el equipo encargado de fabricar el diseño. Por otro lado, el trabajo presentado por Hammond *et al.* (2020) brinda más detalles para esta etapa. Lo más relevante de esta publicación es que a cada diseño le aplica transformaciones para simular la posible contracción o dilatación de estos al fabricarse. Con lo anterior hace un intento de detectar estos dos errores de fabricación desde la etapa de optimización.

3.2 Inconveniente 2: La Optimización

En Malheiros-Silveira y Delalibera (2020) se compararon dos algoritmos en la optimización de un dispositivo. A diferencia de Prosopio-Galarza *et al.* (2019), la comparación no se realizó en base a la cantidad de iteraciones realizadas por cada algoritmo, en cambio se hizo de acuerdo a la cantidad de simulaciones realizadas (≈ 2000), es decir, la cantidad de veces que se evaluó la FOM. Esta estrategia es más adecuada para comparar algoritmos con distintas características de una manera más justa.

En otros trabajos que se centran en comparar algoritmos para optimizar un mismo dispositivo se sigue la misma idea para la comparación (Gregory *et al.*, 2015, Schneider *et al.*, 2019); sin embargo, no parece haber un consenso sobre algún algoritmo que funcione bien para optimizar cualquier dispositivo.

En general, PSO y GA han sido extensamente usados en el área de acuerdo de acuerdo a reseñas como las de Elsawy *et al.* (2020) y Campbell *et al.* (2019). Tal y como es señalado en estos trabajos, el desempeño de ambos algoritmos es sensible a los parámetros escogidos. Esta es un gran inconveniente debido a que escoger parámetros adecuados

puede consumir mucho tiempo y esto se debe hacer independientemente para cada dispositivo. Con el propósito de superar esta dificultad, distintos trabajos están optando por usar algoritmos que no necesiten configurar parámetros internos.

Bajo este enfoque, en Gregory *et al.* (2015) se resaltó el buen desempeño que pude obtener el algoritmo CMA-ES en la optimización de ciertos dispositivos, llegando a superar al PSO. De manera similar, en Schneider *et al.* (2019) se realizó un trabajo muy completo comparando distintos algoritmos llegando a resultados donde la optimización bayesiana mostró los resultados más prometedores. Y, aunque no entra en mucho detalle sobre la razón de esta elección, en Su *et al.* (2020) se empleó el algoritmo L-BFGS-B en la optimización de un *bend* y WDM llegando a obtener resultados destacados.

Un aspecto importante a resaltar de estos últimos tres trabajos es que al comparar distintos algoritmos para optimizar dispositivos fotónicos necesitamos contar con (i) un elevado número de simulaciones y (ii) distintas ejecuciones que comiencen con diferentes puntos iniciales.

Como se ha discutido en este capítulo, la parametrización de nuestros dispositivos usando un bajo número de parámetros puede ir condicionando nuestros resultados, ante ellos podemos realizar una parametrización más flexible basada en píxeles. Esto supone nuevos desafíos, mas ya existen estrategias para afrontarlos e incluso para incluir restricciones de fabricación a esta parametrización. Por otro lado, aunque no se mencionó explicítamente, no parece haber muchas investigaciones que comparen distintos algoritmos cuando trabajamos con un elevado número de parámetros para representar nuestros dispositivos, en especial si los algoritmos no son PSO o GA.

Capítulo 4

Metodología

En el presente capítulo se describe la metodología a seguir en esta tesis. Primero, se describe de forma general los seis pasos que se trabajaran. Luego, se detalla en una sección completa cada uno de estos pasos. Finalmente, se brindan los alcances y limitaciones de la propuesta.

Como se muestra en la Figura 4.1, en esta investigación se siguieron los siguientes seis pasos: (i) preparación de simulación, (ii) preparación de optimización, (iii) optimización continua, (iv) optimización discreta, (v) optimización de fabricación y (vi) preparación para fabricación.

Estos pasos se siguieron para optimizar tanto un *bend* como un WDM. Una vez preparada la simulación, la etapa de optimización (continua, discreta y de fabricación) se realizó tres veces por cada algoritmo. La primera ejecución usa un valor de semilla de 128, la segunda de 256 y la tercera de 512. De esta manera se aseguró iniciar con diseños aleatorios y mantener los resultados reproducibles.

Es importante señalar que los resultados de la optimización continua se usan como punto de inicio para la optimización discreta. Asimismo, los resultados de la optimización discreta se utilizan como entrada para la optimización de fabricación. Este proceso tiene como fin el mantener un buen resultado (Yang y Fan, 2017).

En particular, el valor de β, presente en la Ecuación 2.12, representa el factor discretizador de nuestros diseños. Este valor se va incrementando en la optimización discreta

y de fabricación como se muestra en la Figura 4.1. Finalmente, tras haber conseguido diseños optimizados se realizó un post-procesamiento para dejar los diseños en un formato listo para fabricación.

En las siguientes subsecciones se explica en detalle cada una de los pasos de la metodología seguida en esta tesis.

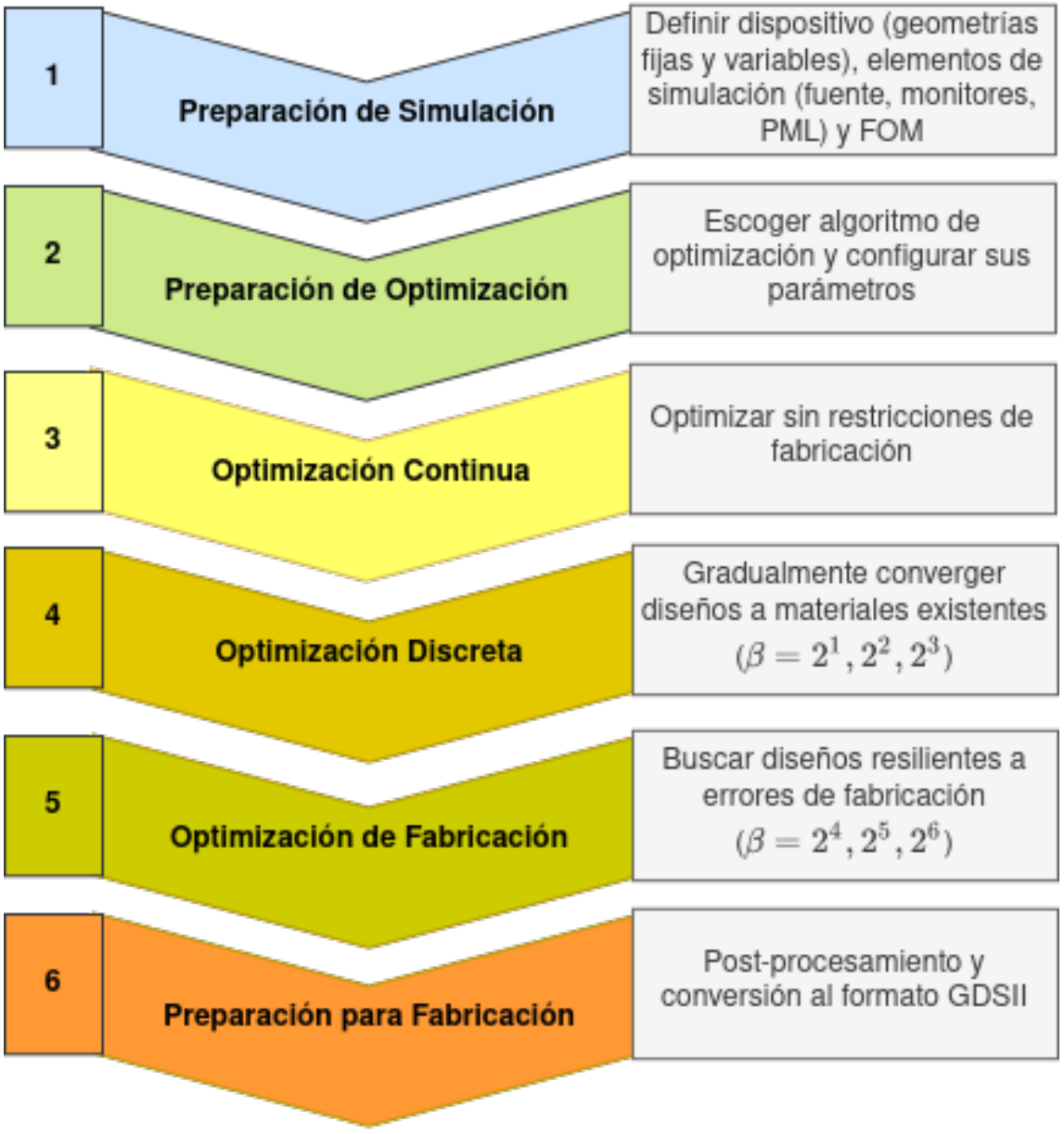


FIGURA 4.1: Metodología del trabajo de investigación

4.1 Preparación de Simulación

El *bend* y WDM se parametrizaron usando la parametrización basada en píxeles descrita en la Sección 2.2. La implementación se realizó en SPINS. La descripción detallada para los dos dispositivos de estudio se presentan en las siguientes dos subsecciones.

4.1.1 *Bend*

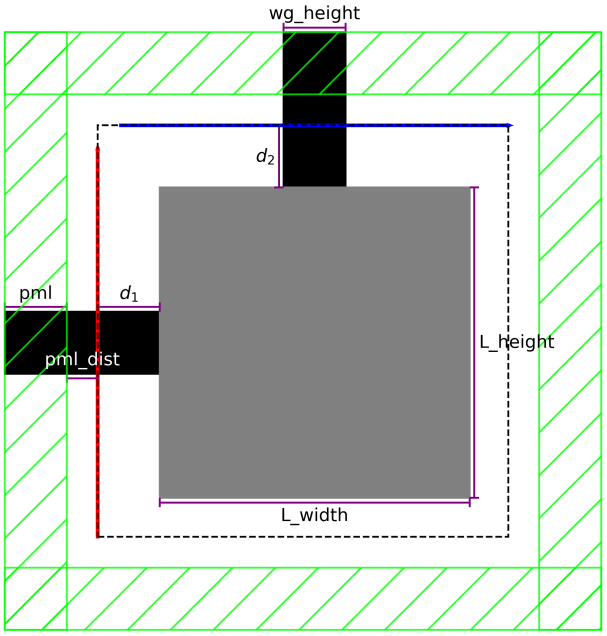


FIGURA 4.2: Parámetros del diseño del *bend* a optimizar.

En la Figura 4.2 se muestra el diseño y parámetros del *bend* a utilizar. Los rectángulo negros representan las guías de onda, estos tienen las mismas dimensiones aunque en distinta orientación. La región gris representa la región de diseño. La recta roja simula la fuente y la recta azul el monitor, su extensión está simbolizada por las variables *source* y *monitor*, respectivamente. Además, la profundidad de estos dos elementos es la misma,

valor denotado como *z_length*. Por otro lado, la región punteada se utiliza como referencia para definir el PML (región verde), del cual dista un valor definido como *pml_dist*. La profundidad de toda la geometría es especificada por la variable *depth*.

Cabe destacar que los parámetros fueron escogidos inspirados por el trabajo de Su *et al.* (2020). De este modo, la región de diseño se dividió en rectángulos de $16nm \times 16nm$, obteniendo así una matriz de 125×125. El valor de los demás parámetros se presenta en la Tabla 4.1.

Parámetro	Valor (nm)
wg_height	400
pml	300
pml_dist	200
d_1	400
d_2	400
L_width	2000
L_height	2000
source	2500
monitor	2500
z_length	1000
depth	220

TABLA 4.1: Parámetros usados en el diseño del *bend* a optimizar.

4.1.2 WDM

En la Figura 4.3 se muestra el diseño y parámetros del WDM a utilizar. Los rectángulo negros representan las guías de onda, estos tienen las mismas dimensiones. La región gris representa la región de diseño. La recta roja simula la fuente y su extensión se simboliza por la variable *source*. Las rectas azul y marrón describen los monitores cuya extensión está definida por las variables *monitor*. Además, la profundidad de la fuente y monitores es la misma, denotado como *z_length*. Por otro lado, la región punteada se utiliza como referencia para definir el PML (región verde), del cual dista un valor definido como *pml_dist*. La profundidad de toda la geometría es especificada por la variable *depth*.

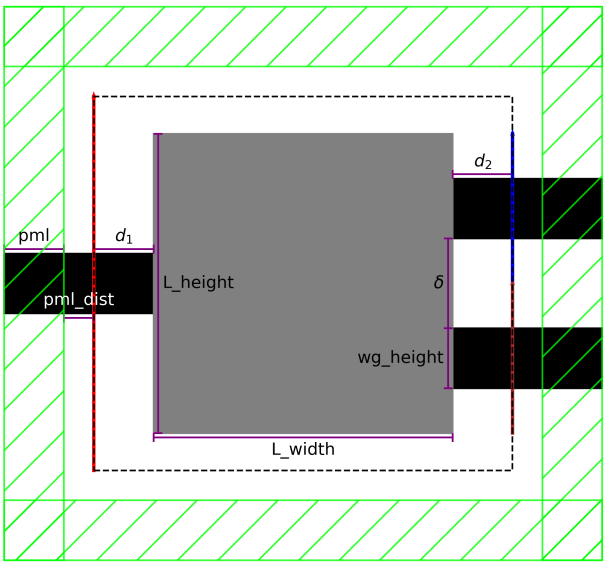


FIGURA 4.3: Parámetros del diseño del WDM a optimizar.

En este caso, los parámetros fueron escogidos inspirados por el trabajo de Christiansen y Sigmund (2021b). Del mismo modo como se realizó para el *bend*, la región de diseño se dividió en rectángulos de $16nm \times 16nm$, obteniendo así una matriz de 125×125. El valor de los demás parámetros se presenta en la Tabla 4.2.

Parámetro	Valor (nm)
wg_height	400
pml	300
pml_dist	200
d_1	400
d_2	400
L_width	2000
L_height	2000
δ	600
source	2500
monitor	1000
z_length	1000
depth	220

TABLA 4.2: Parámetros usados en el diseño del WDM a optimizar.

4.2 Preparación de Optimización

Con lo descrito en la anterior sección ya podemos evaluar distintos diseños. Ahora, utilizando como función objetivo la Ecuación 2.2 para el *bend* y la Ecuación 2.3 para el WDM, estamos ante un problema de optimización. Para resolverlo, la propuesta inicial era utilizar estos cinco algoritmos: (i) GA, (ii) PSO, (iii) CMA-ES, (iv) L-BFGS-B, (v) MMA.

La elección de estos fue por diversos motivos. GA y PSO se escogieron por ser populares en el área (Elsawy *et al.*, 2020, Molesky *et al.*, 2018, Prosopio-Galarza *et al.*, 2019). CMA-ES por haber tenido un buen desempeño en Gregory *et al.* (2015) y las buenas referencias brindadas en Campbell *et al.* (2019). L-BFGS-B por haber obtenido buenos resultados en trabajos como Su *et al.* (2020). MMA por haber sido recomendado como un buen candidato para la optimización topológica (Christiansen y Sigmund, 2021b).

Por trabajos como los de Su *et al.* (2020) y Christiansen y Sigmund (2021b) ya nos podíamos anticipar que al usar L-BFGS-B o MMA se obtendrían buenos resultados en parte gracias a usar la gradiente para guiar la búsqueda. Sin embargo, por la cantidad de variables utilizadas (125×125) no podíamos asegurar lo mismo con GA, PSO y CMA.

Debido a ello, primero se realizó un experimento preliminar para evaluar el desempeño de estos algoritmos al realizar como máximo 1200 evaluaciones en la optimización del *bend* propuesto, estos resultados son presentados en la Figura 4.4. Como podemos observar, especialmente del CMA-ES, hay una ligera tendencia de mejora. Por otro lado, por los resultados mostrados en Christiansen y Sigmund (2021a) uno esperaría que algoritmos como GA lograran obtener resultados similares a los obtenidos por algún algoritmo basado en la gradiente (e.g. L-BFGS-B y MMA); sin embargo, esto podría necesitar evaluar la función objetivo un número muy elevado de veces (entre 10^4 y 10^5).

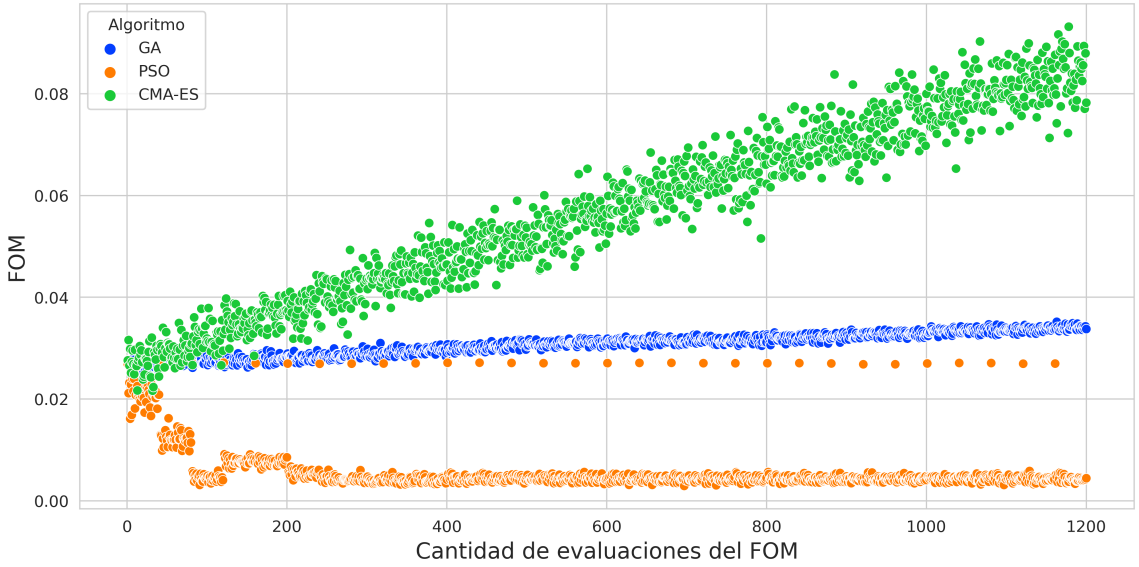


FIGURA 4.4: Comparación del desempeño de GA, PSO y CMA-ES para optimizar el *bend* propuesto.

De este modo, en esta tesis se optó por comparar solamente algoritmos de primer orden, es decir, que utilicen el cómputo de la gradiente en sus rutinas. Así, se seguirá evaluando los algoritmos GA, PSO y CMA-ES pero ahora en sus versiones que usan la gradiente. Para poder comparar el desempeño de estos se configuró la máxima cantidad de veces que podían evaluar la función objetivo. Los parámetros específicos con los que se ejecutaron cada algoritmo se detallan en la Tabla 4.3 siguiendo la nomenclatura presentada en la Sección 2.5.

Es resumen, usando simulaciones en 3D FDFD con SPINS, evaluaremos estos cinco algoritmos: (i) G-GA, (ii) G-PSO, (iii) G-CMA-ES, (iv) L-BFGS-B, (v) MMA.

4.3 Optimización Continua

En esta etapa se configuró la máxima cantidad de evaluaciones a 2000. Luego, se ejecutaron los cinco algoritmos para optimizar el *bend* y WDM. Cada algoritmo se ejecutó 3 veces, la primera con un valor de semilla de 128, la segunda de 256 y la tercera de 512.

Algoritmo	Parámetro	Valor (nm)
G-GA	population_size	40
G-GA	n_selected_parents	10
G-GA	range	0.1
G-GA	prob_mutation	0.1
G-PSO	population_size	40
G-PSO	range	0.1
G-PSO	ω	0.5
G-PSO	c_1	0.5
G-PSO	c_2	0.5
G-CMA-ES	population_size	40
G-CMA-ES	σ	0.3

TABLA 4.3: Parámetros usados por los algoritmos de optimización.

La idea de esta etapa es optimizar los diseños sin imponer ninguna restricción. Es decir, directamente usamos la parametrización $\boldsymbol{P}$ para evaluar la permitividad ε (Ecuación 2.4), lo cual lo podemos representar como $(\boldsymbol{P} \rightsquigarrow \varepsilon)$.

4.4 Optimización Discreta

Los objetivos de esta etapa son dos: (i) obtener diseños con un mínimo radio de curvatura r_f e (ii) ir eliminando las regiones grises. Para el primer objetivo se aplicó la Ecuación 2.13 a la parametrización $\boldsymbol{P}$ usando un radio de curvatura $r_f = 80nm$. Como $80nm$ es igual a 5 veces el tamaño de los píxeles configurados en la Sección 4.1, solamente fue necesario explorar una submatriz de 11×11 alrededor de cada elemento para el cálculo de esta ecuación. Para el segundo objetivo se aplicó la Ecuación 2.12 al resultado conseguido por el anterior filtro. Este proceso lo podemos representar como $(\boldsymbol{P} \rightsquigarrow \widetilde{\boldsymbol{P}} \rightsquigarrow \widetilde{\widetilde{\boldsymbol{P}}} \rightsquigarrow \varepsilon)$.

Para la optimización se utilizó los resultados de la optimización continua como punto inicial. Para ello se realizó una primera optimización configurando la máxima cantidad de evaluaciones en 800 y aplicando a cada diseño los filtros descritos anteriormente con $\beta = 2^1$ y $\eta = 0.5$ para la Ecuación 2.12. El resultado de esta optimización se usa

como punto inicial para repetir este proceso, pero ahora con $\beta = 2^2$. Finalmente, se repite una vez más el procedimiento con $\beta = 2^3$.

4.5 Optimización de Fabricación

Por último, se busca imponer a los diseños restricciones de fabricación y robustez a posibles errores de fabricación.

Tomando como referencia el trabajo de Hammond *et al.* (2020) para asegurar un buen desempeño pese a los errores de fabricación, por cada parametrización $\boldsymbol{P}$ se calculó:

- $\widetilde{\widetilde{\boldsymbol{P}}}_d$ que representa el diseño como si el dispositivo se hubiera dilatado.
- $\widetilde{\widetilde{\boldsymbol{P}}}_i$ que representa el diseño normal.
- $\widetilde{\widetilde{\boldsymbol{P}}}_e$ que representa el diseño como si el dispositivo se hubiera expandido.

Estos tres elementos se calcularon siguiendo los mismos filtros descritos en la anterior sección ($\boldsymbol{P} \rightsquigarrow \widetilde{\boldsymbol{P}} \rightsquigarrow \widetilde{\widetilde{\boldsymbol{P}}} \rightsquigarrow \varepsilon$). Pero, para el diseño dilatado se usó $\eta = 0.55$, para el diseño normal $\eta = 0.5$ y para el diseño expandido $\eta = 0.45$.

Luego, se definió una nueva función objetivo mediante la siguiente ecuación:

$$F_{obj} = max(min(f_{obj}(\widetilde{\widetilde{\boldsymbol{P}}}_d), f_{obj}(\widetilde{\widetilde{\boldsymbol{P}}}_i), f_{obj}(\widetilde{\widetilde{\boldsymbol{P}}}_e)) \tag{4.1}$$

Así, considerando el peor escenario ante dos posibles errores de fabricación (expansión o contracción), se busca el diseño más resiliente posible. Para optimizar esta nueva función objetivo se realizó el mismo proceso de la optimización discreta, pero ahora con valores $\beta = 2^4, 2^5, 2^6$. De esta manera, en paralelo, seguimos con el proceso de discretizar nuestros resultados.

4.6 Preparación para Fabricación

Para que nuestros diseños puedan ser fabricados, necesitamos representarlo en formato GDSII. Para ello, con los mejores resultados de cada algoritmo usamos la distribución de la permitividad asociada a su parametrización, $\epsilon(p)$, para obtener la geometría del dispositivo. Así, nuestra tarea se redujo a aproximar el contorno de esta geometría con polígonos para luego realizar la conversión al formato GDSII. Para ello se uso el paquete *matplotlib* para dibujar la geometría y obtener un polígono que la aproxime.

4.7 Alcances y Limitaciones

El presente trabajo solo se realizo mediante simulaciones computacionales, no se llegó a la parte de fabricación por temas de tiempo y dinero.

En este capitulo hemos explicado los pasos a seguir en esta tesis. Con estos seis pasos responderemos a los objetivos planteados.

REFERENCIAS BIBLIOGRÁFICAS

Anderson, E., González, J., Gazman, A., Azevedo, R., and Bergman, K. (2018). Optically connected and reconfigurable gpu architecture for optimized peer-to-peer access. In *Proceedings of the International Symposium on Memory Systems*, MEMSYS '18, page 257–258, New York, NY, USA. Association for Computing Machinery.

Bogaerts, W. and Chrostowski, L. (2018). Silicon photonics circuit design: Methods, tools and challenges. *Laser & Photonics Reviews*, 12(4):1700237.

Campbell, S. D., Sell, D., Jenkins, R. P., Whiting, E. B., Fan, J. A., and Werner, D. H. (2019). Review of numerical optimization techniques for meta-device design [invited]. *Opt. Mater. Express*, 9(4):1842–1863.

Christiansen, R. E. and Sigmund, O. (2021a). Compact 200 line matlab code for inverse design in photonics by topology optimization: tutorial. *J. Opt. Soc. Am. B*, 38(2):510–520.

Christiansen, R. E. and Sigmund, O. (2021b). Inverse design in photonics by topology optimization: tutorial. *J. Opt. Soc. Am. B*, 38(2):496–509.

Elsawy, M. M., Lanteri, S., Duvigneau, R., Fan, J. A., and Genevet, P. (2020). Numerical Optimization Methods for Metasurfaces. *Laser and Photonics Reviews*, 14(10):1–17.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. (2015). Qualitatively characterizing neural network optimization problems. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Gregory, M. D., Martin, S. V., and Werner, D. H. (2015). Improved Electromagnetics Optimization. *IEEE Antennas and Propagation Magazine*, 57(june):48–59.

Hammond, A. M., Oskooi, A., Johnson, S. G., and Ralph, S. E. (2020). Robust topology optimization of foundry-manufacturable photonic devices: An open-source fdtd toolbox. In *Frontiers in Optics / Laser Science*, page FTh1C.4. Optical Society of America.

Hansen, N. (2016). The CMA Evolution Strategy: A Tutorial.

Huang, H. and Ouyang, Z. (2018). General method for eliminating wave reflection in 2d photonic crystal waveguides by introducing extra scatterers based on interference cancellation of waves. *Optics Communications*, 406:260–270. Optoelectronics and Photonics Based on Two-dimensional Materials.

Hughes, T. W. and Fan, S. (2016). Plasmonic circuit theory for multiresonant light funneling to a single spatial hot spot. *Nano Letters*, 16(9):5764–5769.

Kochenderfer, M. J. and Wheeler, T. A. (2019). *Algorithms for Optimization*. The MIT Press.

Kudyshev, Z. A., Kildishev, A. V., Shalaev, V. M., and Boltasseva, A. (2020). Machine learning-assisted global optimization of photonic devices. *Nanophotonics*, 10(1):371–383.

Li, N., Ho, C. P., Wang, I.-T., Pitchappa, P., Fu, Y. H., Zhu, Y., and Lee, L. Y. T. (2021). Spectral imaging and spectral lidar systems: moving toward compact nanophotonics-based sensing. *Nanophotonics*, 10(5):1437–1467.

Lukas Chrostowski (2010). *Silicon Photonics Design: From Device to System*.

Malheiros-Silveira, G. N. and Delalibera, F. G. (2020). Inverse design of photonic structures using an artificial bee colony algorithm. *Applied Optics*, 59(13):4171.

Malkiel, I., Mrejen, M., Nagler, A., Arieli, U., Wolf, L., and Suchowski, H. (2018). Plasmonic nanostructure design and characterization via Deep Learning. *Light: Science and Applications*, 7(1).

Molesky, S., Lin, Z., Piggott, A. Y., Jin, W., Vucković, J., and Rodriguez, A. W. (2018). Inverse design in nanophotonics. *Nature Photonics*, 12(11):659–670.

Oskooi, A. F., Roundy, D., Ibanescu, M., Bermel, P., Joannopoulos, J. D., and Johnson, S. G. (2010). Meep: A flexible free-software package for electromagnetic simulations by the FDTD method. *Computer Physics Communications*, 181(3):687–702.

Piggott, A. Y., Petykiewicz, J., Su, L., and Vučković, J. (2017). Fabrication-constrained nanophotonic inverse design. *Scientific Reports*, 7(1):1–7.

Prosopio-Galarza, R., De La Cruz-Coronado, J., Hernandez-Figueroa, H. E., and Rubio-Noriega, R. (2019). Comparison between optimization techniques for Y-junction devices in SOI substrates. *Proceedings of the 2019 IEEE 26th International Conference on Electronics, Electrical Engineering and Computing, INTERCON 2019*, pages 1–4.

Schneider, P. I., Garcia Santiago, X., Soltwisch, V., Hammerschmidt, M., Burger, S., and Rockstuhl, C. (2019). Benchmarking Five Global Optimization Approaches for Nano-optical Shape Optimization and Parameter Reconstruction. *ACS Photonics*, 6(11):2726–2733.

Shen, Y., Harris, N. C., Skirlo, S., Prabhu, M., Baehr-Jones, T., Hochberg, M., Sun, X., Zhao, S., Larochelle, H., Englund, D., and Soljačić, M. (2017). Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 11(7):441–446.

Shen, Y., Meng, X., Cheng, Q., Rumley, S., Abrams, N., Gazman, A., Manzhosov, E., Glick, M. S., and Bergman, K. (2019). Silicon photonics for extreme scale systems. *J. Lightwave Technol.*, 37(2):245–259.

Song, W. and Xie, K. (2008). Optimal design of a multi-mode interference splitter based on SOI. *Optoelectronics Letters*, 4(2):92–95.

Su, L., Piggott, A. Y., Sapra, N. V., Petykiewicz, J., and Vučković, J. (2018). Inverse Design and Demonstration of a Compact on-Chip Narrowband Three-Channel Wavelength Demultiplexer. *ACS Photonics*, 5(2):301–305.

Su, L., Vercruysse, D., Skarda, J., Sapra, N. V., Petykiewicz, J. A., and Vučković, J. (2020). Nanophotonic inverse design with SPINS: Software architecture and practical considerations. *Applied Physics Reviews*, 7(1).

Vuckovic, J. (2019). From inverse design to implementation of practical (quantum) photonics (Conference Presentation). In Soci, C., Sheldon, M. T., and Agio, M., editors, *Quantum Nanophotonic Materials, Devices, and Systems 2019*, volume 11091. International Society for Optics and Photonics, SPIE.

Yang, J. and Fan, J. A. (2017). Topology-optimized metasurfaces: impact of initial geometric layout. *Opt. Lett.*, 42(16):3161–3164.

Zhang, J., Bi, S., and Zhang, G. (2021). A directional Gaussian smoothing optimization method for computational inverse design in nanophotonics. *Materials and Design*, 197:109213.