UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

GRADO EN INGENERÍA EN ELECTRÓNICA Y AUTOMÁTICA INDUSTRIAL

Trabajo Fin de Grado

"Desarrollo de una herramienta de análisis óptico de capas delgadas para la optimización de la eficiencia en células solares"

2015 Alejandro Villalba Sierra

UNIVERSIDAD DE ALCALÁ
Escuela Politécnica Superior

GRADO EN INGENIERIA EN ELECTRONICA Y AUTOMATICA INDUSTRIAL

Trabajo Fin de Grado
Desarrollo de una herramienta de análisis óptico de capas delgadas, para la optimización de la eficiencia en células solares.

Autor: Alejandro Villalba Sierra

Directores: Fernando B. Naranjo Vega (UAH) y
Susana María Fernández Ruano (CIEMAT)

TRIBUNAL:

Presidente: Ernesto Martín Gorostiza

Vocal 1º: Pablo Ramos Sainz

Vocal 2º: Fernando B. Naranjo Vega

CALIFICACIÓN: ……
FECHA: ……

AGRADECIMIENTOS

A mi familia, por el apoyo y ayuda incondicional que me habéis dado a lo largo toda esta etapa.

A Marta Llarandi por ser la fuente de mi esfuerzo y trabajo.

A mis profesores y compañeros, por compartir vuestros conocimientos y por la ayuda que de vosotros he recibido. Especialmente a mi tutor Fernando Naranjo y a mi cotutora Susana María Fernández, por el gran esfuerzo que han hecho, y por ayudarme enormemente a realizar este proyecto, gracias.

Índice de contenido

INDICE DE FIGURAS

RESUMEN EN CASTELLANO

En este trabajo se ha desarrollado de una herramienta basada en Matlab para el análisis de espectros ópticos de transmisión de láminas delgadas. La herramienta se empleará para determinar las propiedades ópticas lineales de dichas láminas, tales como el índice de refracción y el coeficiente de absorción, a través de un ajuste matemático empleando modelos físicos y fenomenológicos a las medidas experimentales. De este modo, los resultados que se obtengan con dicha herramienta serán de gran utilidad para poder aplicarlos en los diseños de células solares de segunda generación.

ABSTRACT

The main aim of this project is to develop a tool based on Matlab to determine the optical parameters of thin films from measured optical spectra. The tool will estimate thin film optical properties, using a mathematical fit to the experimental measurements. Thus, results obtained with the program will allow the known of their main optical parameters in order to be used as component in second generation solar cells.

RESUMEN EXTENDIDO

La caracterización óptica de capas ventana se realiza fundamentalmente midiendo el espectro de transmisión de la capa. A partir de este espectro y suponiendo unas relaciones para los parámetros ópticos preestablecidas para este tipo de materiales, se pueden obtener las propiedades ópticas de las capas en función de la longitud de onda. Ahora bien, la obtención de los parámetros ópticos que determinan las propiedades de las capas exige un proceso de ajuste por mínimos cuadrados de los datos experimentales a una función prueba. Este proceso suele ser tedioso y requiere de potentes algoritmos de ajuste a funciones de varios parámetros. Es por tanto de inmediata aplicación el desarrollo de una herramienta para la realización de estos ajustes con una apropiada interfaz de usuario, de forma que se pueda utilizar en entornos de investigación y desarrollo sin necesidad de conocimientos de algoritmos de ajuste y programación. El entorno Matlab reúne los requisitos para el desarrollo de dicha herramienta, ya que ofrece potentes librerías de ajuste que permiten la definición de parámetros y métodos, así como una interfaz de usuario (GUI) para su uso eficiente y rápido.

Por lo tanto, en este trabajo se desarrollará dicha herramienta teniendo en cuenta los modelos ópticos más comúnmente utilizados en este tipo de materiales, sus parámetros más importantes así como una interfaz de usuario para realizar las tareas de ajuste, introducción de parámetros de partida, selección del modelo a emplear y manejos de ficheros y resultados para su posterior análisis.

La herramienta desarrollada se aplicará directamente a las diferentes láminas de óxidos conductores transparentes fabricadas en el CIEMAT. En concreto, a una serie de muestras en los que se haya variado un parámetro del depósito de las capas. Pare ello se realizarán las medidas ópticas sobre el material y se estimarán los parámetros ópticos más importantes a partir de éstas. En este punto cabe señalar que el depósito del material queda fuera de este trabajo, no así la caracterización óptica de las capas, para lo que se empleará un espectrofotómetro automatizado.

Este proceso permitirá advertir los errores en la propia herramienta en desarrollo y solucionarlos. Además, permitirá estimar

las capas más apropiadas para aumentar la eficiencia de las células solares en desarrollo.

1. MEMORIA

1.1. INTRODUCCIÓN

En este capítulo se describe la célula solar y los diferentes tipos existentes en la actualidad. Se comenta una breve historia de este dispositivo, así como también se realiza un repaso a la teoría básica que lo gobierna, y los parámetros ópticos que lo caracterizan. En el capítulo 1.2 se detallará el desarrollo del software, para lograr un entendimiento del conjunto de este trabajo, para finalizar con las conclusiones y el trabajo futuro.

1.1.1. Evolución de la energía solar fotovoltaica

Sensor es aquel dispositivo electrónico capaz de convertir una magnitud física en una magnitud eléctrica. En este sentido, una célula solar se encarga de convertir la energía procedente de la luz solar en energía eléctrica, a través del efecto fotoeléctrico.

La energía solar fotovoltaica, entre otras fuentes renovables, se presenta como una alternativa a las formas tradicionales de producción de energía eléctrica. Fue en la década de los 60 cuando empezó a desarrollar esta tecnología, a partir, entre otras motivaciones, por la crisis del petróleo y la llegada de la era espacial. Pero aún no ha satisfecho las expectativas que la rodean.

Este tipo de fuente de energía presenta grandes ventajas asociadas a su inagotable y bien distribuido "combustible": la energía solar, en la (figura 1.1) podemos ver la distribución de la radiación solar en todo el planeta "*Radiación solar, jueves, 16 de noviembre de 2012. Consulta: Jueves, 10 de septiembre de 2015 Disponible en: < http://www.comuexpress.blogspot.com.es/2012/11/radiacion-solar.html>*". Lo que es cierto es que la energía solar, 60 años después del inicio de su desarrollo, sigue siendo una enorme promesa, aunque cada vez más real.

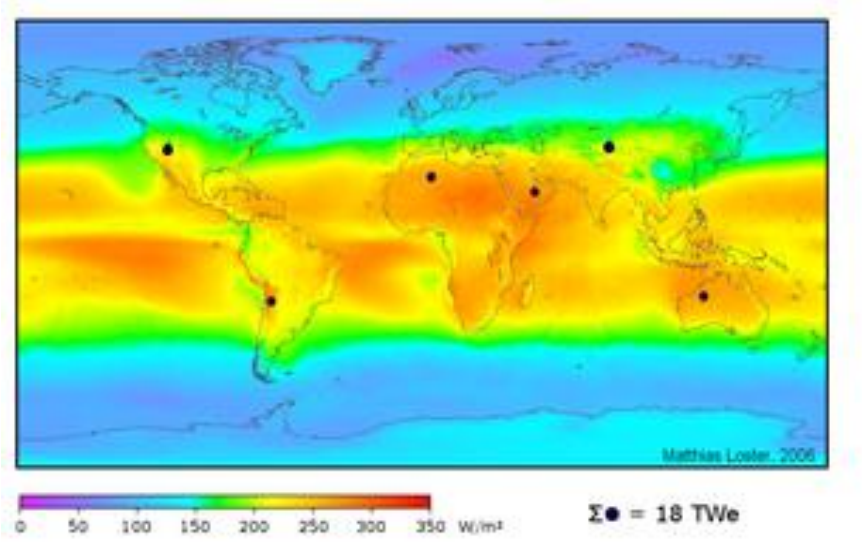


Figura 1.1 Distribución de la radiación solar

A lo largo de los años el aprovechamiento de la misma desde el punto de vista económico, no ha sido su fuerte. Así a pesar de su robustez, modularidad y sencillez de aplicación, la energía solar fotovoltaica continúa siendo cara, entre 5 y 10 veces más cara que el coste medio de producción de energía en la Unión Europea 0,04€/kWh "*La Unión Europea y el mercado energético Martes, 18 de febrero de 2014. Consulta: Jueves, 10 de septiembre de 2015 Disponible en: <* *http://www.elcaptor.com/2014/02/la-union-europea-y-el-mercado-energetico.html*>"

A pesar de las diferencias económicas, en los últimos años la evolución del mercado de módulos fotovoltaicos ha crecido casi exponencialmente y de igual manera la producción de energía debida a esta tecnología, lo que demuestra la fortaleza de este "nuevo" mercado. Atendiendo a costes de producción, y tal y como se observa en la Figura 1.2, la tecnología dominante en el mercado de células sigue siendo aquella basada en silicio cristalino.

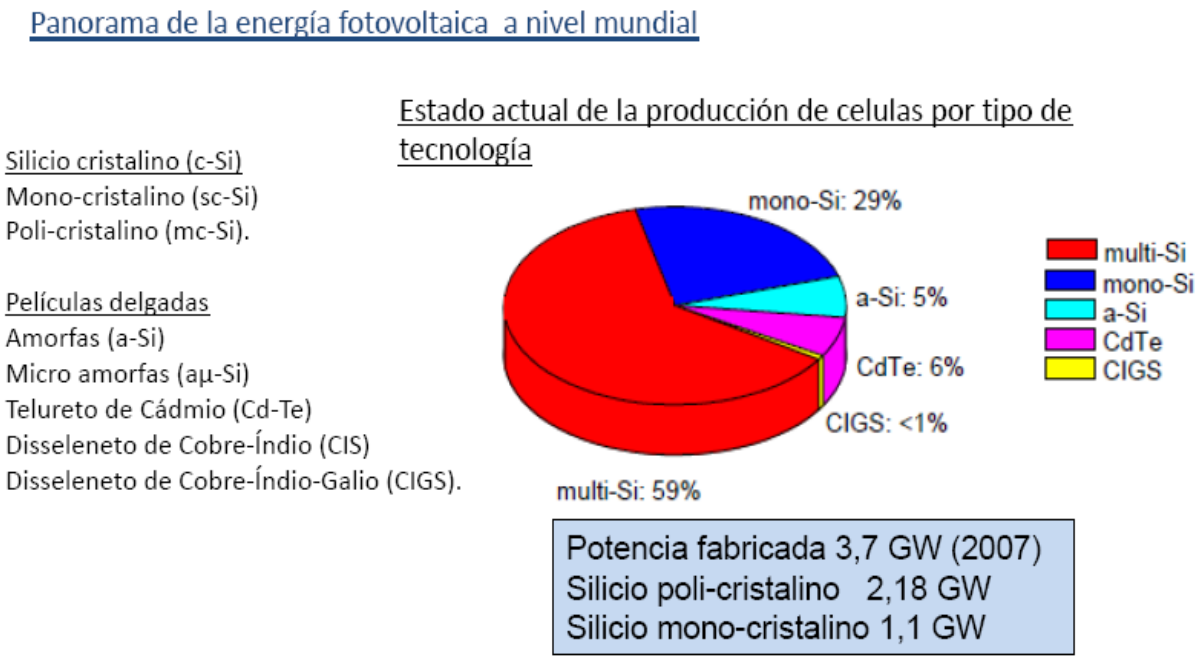


Figura 1.2 **Producción mundial de células solares por tipo de tecnología**

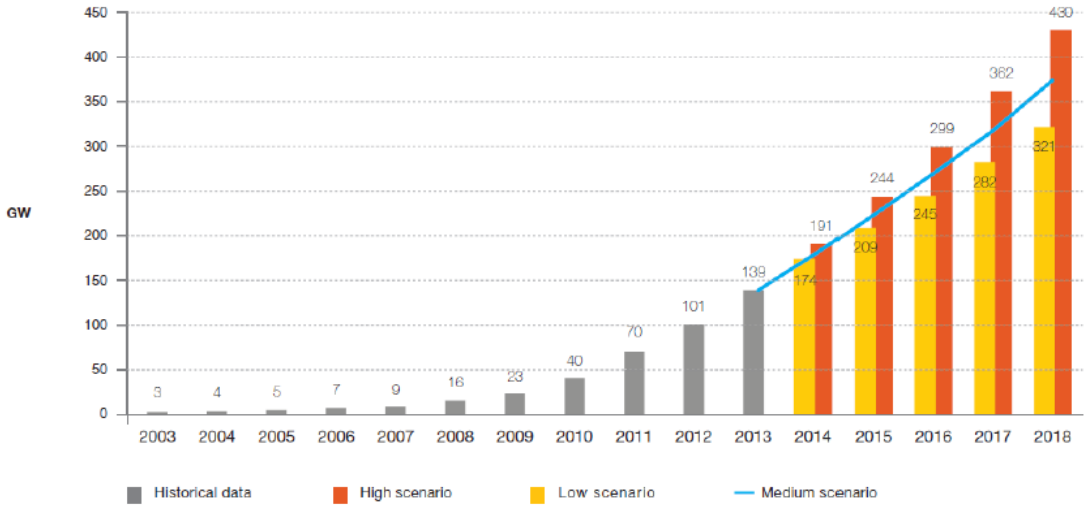


Figura 1.3 **Producción mundial de energía FV**

Como se muestra en la **(figura 1.3)** la generación de energía mundial mediante módulos solares instalados ha sido bastante escasa, pero alberga un gran potencial, dando previsiones de producción, en los años venideros, que multiplican casi por 10 la producción de hace 5 años.

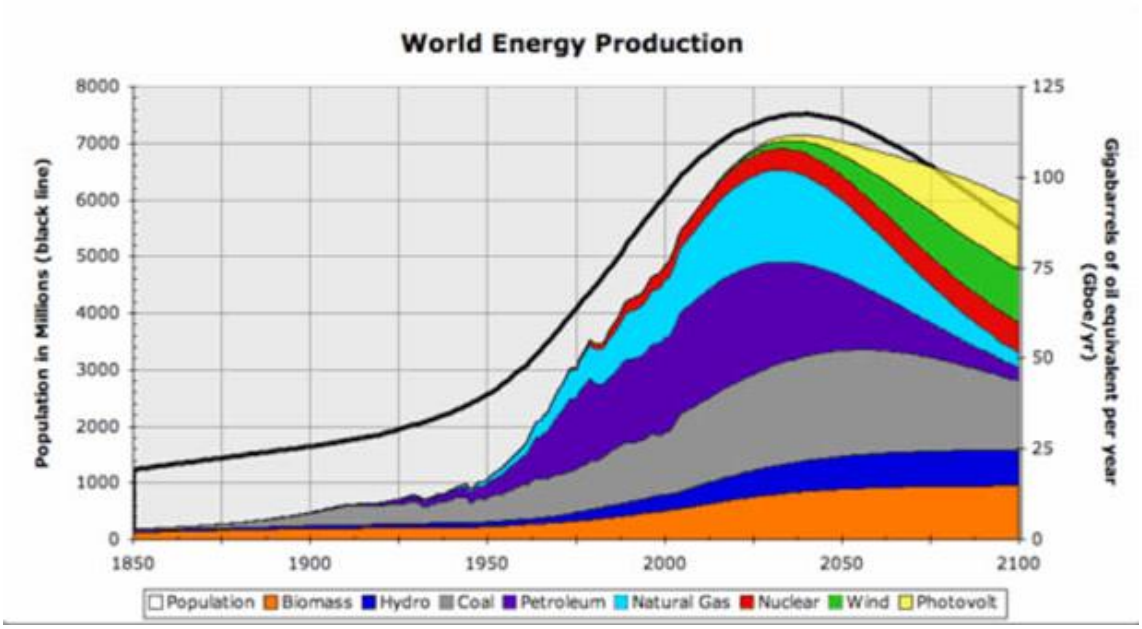


Figura 1.4 Producción mundial de energía

Sin embargo, la energía producida mediante esta tecnología no representa más que un 5% de la producción total de energía a nivel mundial, tal y como se muestra en la Figura 1.4, pero se estima que en 20 años se alcance un porcentaje superior al 15% "*Innovative Clean Energy Group. World Energy production vs Consumption Consulta: Jueves, 10 de septiembre de 2015 Disponible en: <http://www.iceuls.com/present_energy_resources/world_energy_production_vs_consumption.php>*".

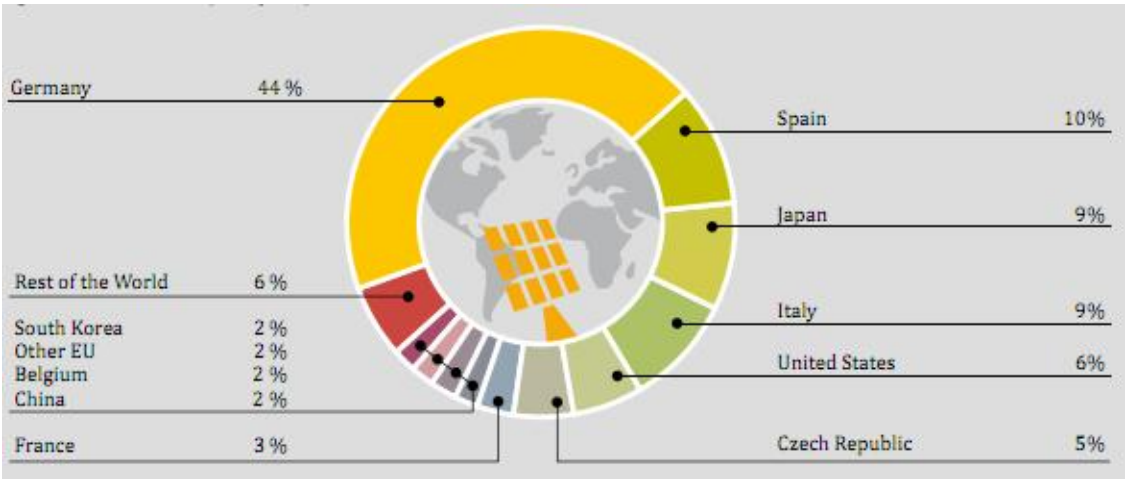


Figura 1.5 Situación de la energía solar fotovoltaica

En cuanto a la localización de los mayores productores de energía fotovoltaica, Europa se sitúa con una amplia diferencia en primera posición, siendo España el segundo productor mundial de energía fotovoltaica, solo por detrás de Alemania (**ver Figura 1.5).** Por esto mismo, éste es un gran mercado en el que invertir.

La primera célula solar se desarrolló en 1883, cuando Charles Fritts recubrió una muestra de selenio semiconductor con oro. En 1946 se registra la primera patente de célula fotovoltaica y es en 1954 cuando se descubre en los laboratorios Bell la sensibilidad a la luz de los semiconductores dopados con impurezas. Tras pocos años después de implementar la primera célula solar de silicio se aumentaron las eficiencias de conversión que oscilaban en torno al 0,5% hasta un 6%. Las mejoras introducidas desde entonces son evidentes. En este punto cabe destacar que el record en eficiencia de conversión en electricidad se ha establecido en un 44.7% utilizando células multi-unión basadas en materiales III-V y que han sido desarrolladas en el Instituto Fraunhofer, "*MERINO, Luis. Nuevo record de eficiencia para una célula solar. Martes, 24 de septiembre de 2013. Consulta: Jueves, 2 de abril de 2015 Disponible en: <http://www.energias-renovables.com/articulo/nuevo-record-de-eficiencia-para-una-celula-20130924>*". En esta tecnología, múltiples uniones p-n de diferentes materiales semiconductores III-V se apilan, de tal forma que cada una absorba diferentes rangos de longitud de onda del espectro solar.

Desde finales de la década de los 50, la carrera espacial influyó considerablemente en el desarrollo de células solares para alimentar eléctricamente a los satélites artificiales. En 1970 se fabrica la primera célula de arseniuro de galio (GaAs) material que dominó la fabricación de células fotovoltaicas hasta la década de los 80. Década a partir de la cual podemos hablar de un crecimiento exponencial en la escena mundial en la instalación de paneles solares.

En la (figura 1.6) podemos ver la evolución, a lo largo de los años, de las distintas tecnologías tanto establecidas como emergentes, que existen actualmente y cuáles son sus valores de eficiencia, estableciéndose una comparativa entre ellas.

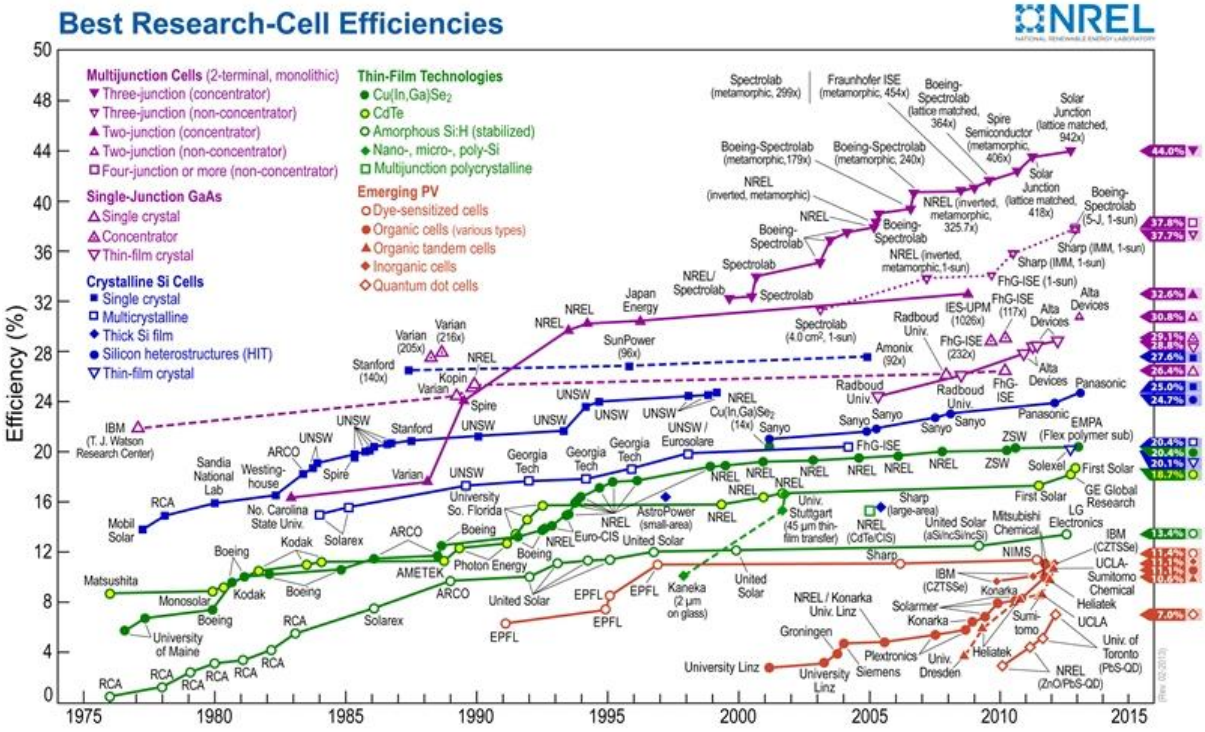


Figura 1.6 **Evolución de la eficiencia por tipo de tecnología**

Una célula solar es un dispositivo electrónico capaz de transformar la energía proveniente de la radiación solar en energía eléctrica, basa su funcionamiento en el principio fotoeléctrico. Está formada por la unión de dos capas de materiales **semiconductores** uno tipo P y otro tipo N, con contactos eléctricos en la parte superior y le inferior. Ayudándose del efecto fotoeléctrico es capaz de generar una corriente continua, cuando los semiconductores son iluminados por un haz de fotones.

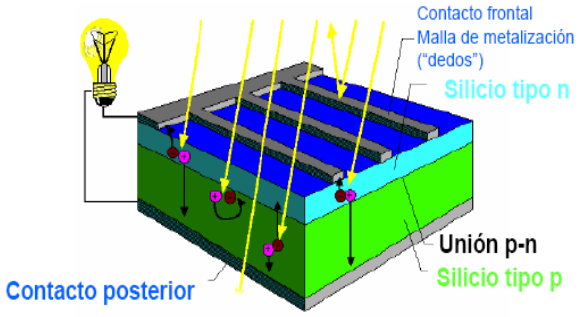


Figura 1.7 Estructura de una célula solar

Para entender bien qué se pretende conseguir con este proyecto, es interesante realizar un repaso a distintos puntos asociados principalmente a los semiconductores y a la fabricación de células solares.

1.1.2. Semiconductores

Los semiconductores son materiales que se comportan como un conductor o como un aislante dependiendo de diversos factores, como por ejemplo el campo eléctrico o magnético, la presión, la radiación que le incide, o la temperatura del ambiente en el que se encuentre.

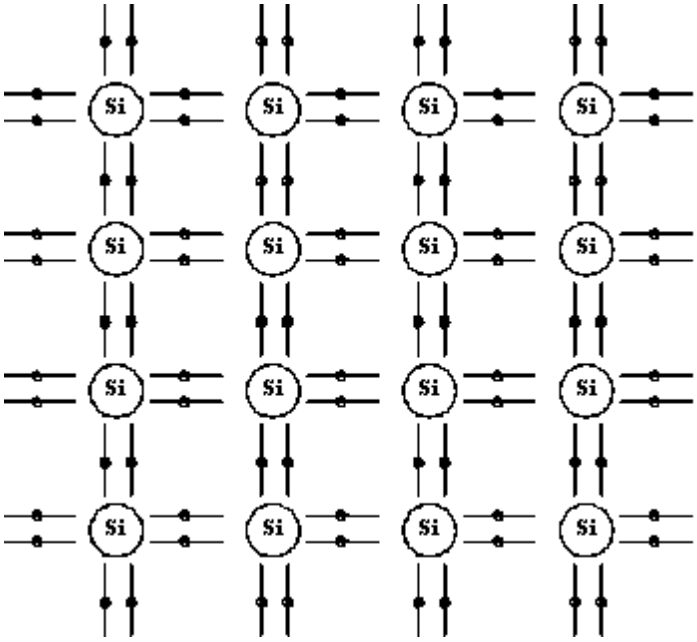


Figura 1.8 Semiconductor intrínseco

.

Los principales elementos que, combinados, pueden comportarse como semiconductores son los siguientes: Cu, Zn, Cd, Al, Ga, In, Si, Ge, N, P, As, Sb, S, Se, Te.

El silicio es el segundo elemento más abundante en la corteza terrestre (después del oxígeno) representando un 27'7% de su peso. El silicio forma parte del grupo 14**,** y tiene valencia IV, por lo que en un enlace metálico o covalente el Silicio compartirá 4 átomos, con el o los elementos que se asocie. Por sus propiedades semiconductoras tiene un interés especial en la industria electrónica como material básico para la formación de obleas que son usadas en la fabricación de transistores, dispositivos semiconductores y por ende células solares.

En el silicio cristalino los átomos están unidos mediante enlaces covalentes, consistente en que cada átomo comparte un electrón con el átomo vecino, por lo que estará rodeado por ocho electrones. Al alcanzar cierta energía el enlace puede romperse, quedando un electrón libre, el cual contribuirá a la generación de una corriente

eléctrica. Por lo tanto podemos hablar de dos niveles distintos de energía:

Uno de baja energía, denominado banda de valencia (Ev), en el cual todos los electrones se encuentran enlazados

Otro de mayor energía que se denomina banda de conducción (Ec), en el que se encuentran los electrones que, debido al aporte de energía, han quedado libres, pudiendo entonces formar parte de una corriente eléctrica.

Entre estos dos niveles de energía no hay ningún estado intermedio. Si la energía es suficientemente alta para producir este salto, el electrón romperá el enlace y pasará de la banda de valencia a la banda de conducción, dejando un estado electrónico libre en la banda de valencia, denominado hueco. La interfase entre la banda de conducción y la banda de valencia se le denomina ancho de banda prohibida del semiconductor, esta es una banda de paso y no un estado intermedio, por lo que el electrón sólo pasará por aquí al producirse el salto, si la energía no es suficientemente alta para que se complete el salto, el electrón permanecerá en la banda de valencia. A la energía necesaria para provocar el salto de un electrón de la banda de valencia a la banda de conducción se le denomina energía de gap (Eg). La **(figura 1.9)** muestra el salto de un electrón de la banda de valencia a la banda de conducción, al superarse la energía de gap, generando un hueco en la banda de valencia.

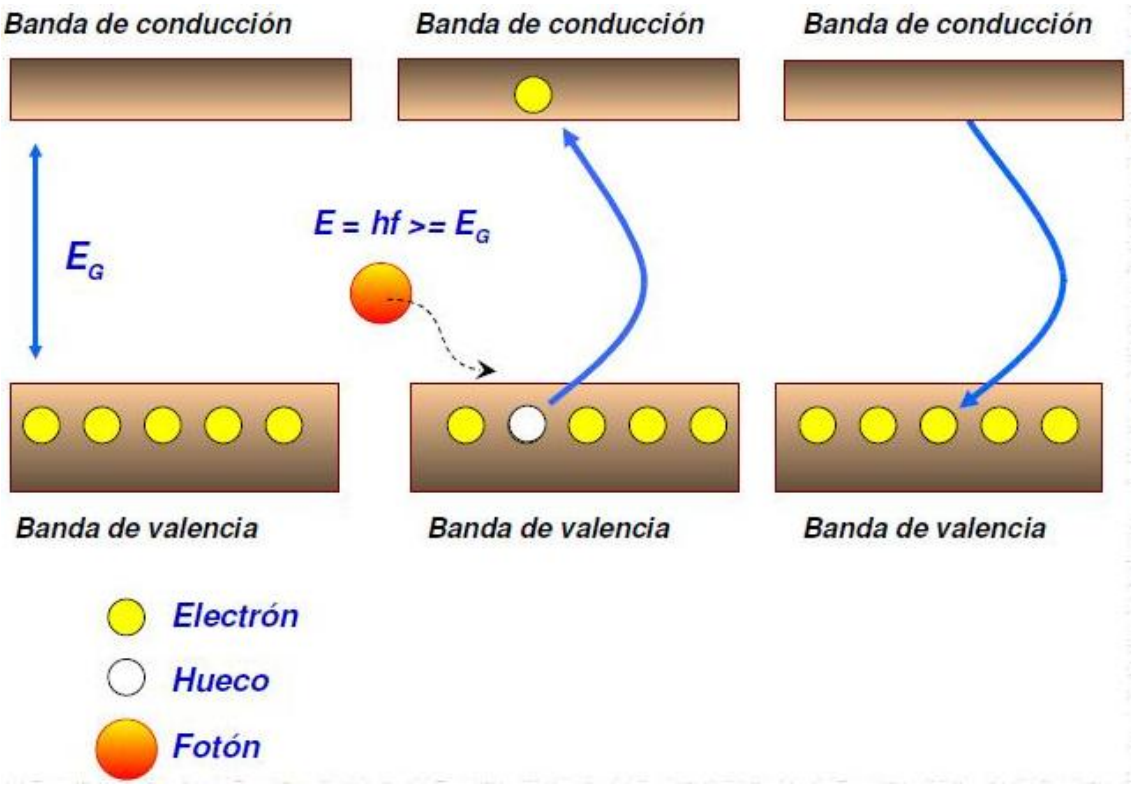


Figura 1.9 Salto de electrón entre bandas

Más adelante se abordará en profundidad este concepto y su gran importancia en la caracterización de materiales semiconductores utilizados en tecnología fotovoltaica. A modo de ejemplo, a temperatura ambiente para el silicio esta energía es de 1,14eV y para el arseniuro de galio (AsGa) es de 1,42eV.

1.1.2.1. Dopaje de los semiconductores

El término dopaje en la ciencia de los materiales hace referencia al proceso intencionado de agregar impurezas a un semiconductor extremadamente puro, con el fin de cambiar sus propiedades eléctricas. Existen dos tipos de materiales dopantes: tipo N y tipo P dependiendo de si en el proceso dopante han aumentado los electrones o los huecos del material.

Se llama semiconductor tipo N al que posee átomos de impurezas, que permiten la aparición de electrones libres sin huecos libres asociados a los mismos. Los átomos de este tipo se llaman donantes ya que "donan" o entregan electrones. Para el silicio serán elementos de valencia 5 como el Fósforo. A diferencia de los átomos que conforman la estructura original, posee un electrón no ligado, por lo tanto la energía necesaria para separar a este electrón será inferior que la necesitada para romper el enlace que se forma en los cristales de

silicio. Finalmente tendremos más electrones que huecos por lo que los primeros serán los portadores mayoritarios y los últimos los minoritarios. La cantidad de portadores mayoritarios será función directa de la cantidad de átomos de impurezas introducidos

Se llama semiconductor tipo P al que tiene más átomos de impurezas que permiten la formación de huecos libres sin que aparezcan, como ocurre al romperse un enlace, los electrones asociados a los mismos. Los átomos de este tipo se llaman aceptadores, ya que "aceptan" o toman un electrón, y serán de valencia tres como el Aluminio, el Boro o el Galio. Debido a que solo tiene tres electrones en su capa de valencia, cuando sustituye a un átomo de silicio aparecerá un enlace libre que tendrá afinidad por tomar electrones de los átomos próximos, generando finalmente más huecos que electrones libres, por lo que los primeros serán los portadores mayoritarios y los segundos los minoritarios. Al igual que en el material N, la cantidad de portadores mayoritarios será función directa de la cantidad de átomos de impurezas producidos.

El principio de funcionamiento de las células solares es, como hemos dicho, el efecto fotovoltaico. Este efecto se produce cuando la radiación solar incide sobre la unión P-N del semiconductor, en este momento se puede generar in par electrón-hueco y el campo eléctrico orienta las cargas del electrón y el hueco, estableciéndose la diferencia de potencial a partir de la cual circula corriente por la carga.

1.1.3. Células solares: Materiales utilizados, tipos y desarrollo

Una célula solar o célula fotovoltaica es un dispositivo que transforma la energía luminosa (fotones) en energía eléctrica (electrones), por medio del denominado efecto fotovoltaico. Se suele denominar célula solar la que usa como energía de partida la luz solar, mientras que la célula fotovoltaica se usa cuando el origen de la luz no está especificado.

Figura 1.10 Célula fotovoltaica comercial.

El efecto fotovoltaico consiste, en sus principios más básicos, en utilizar la capacidad de algunos materiales – semiconductores – en liberar electrones al ser bombardeados con fotones. Como se ha mostrado en los apartados anteriores, el gap de energía necesario para desprender los electrones es menor que la energía aportada por los fotones, de manera que al absorber el fotón, se desprende un electrón, dejando un "hueco" en su lugar.

1.1.3.1. Tipos de células solares

Partiendo de los objetivos principales de la tecnología fotovoltaica, producir electricidad a bajo coste y con mínimos índices de contaminación, se necesitan células solares baratas y eficientes.

Estas células solares se clasifican en tres generaciones y siguen las estructuras de la (figura 1.11), en función del orden en las que pasaron a ser relevantes. En la actualidad, hay investigación de manera concurrente en las tres generaciones, pero las tecnologías de primera generación son las que están más representadas en la producción industrial.

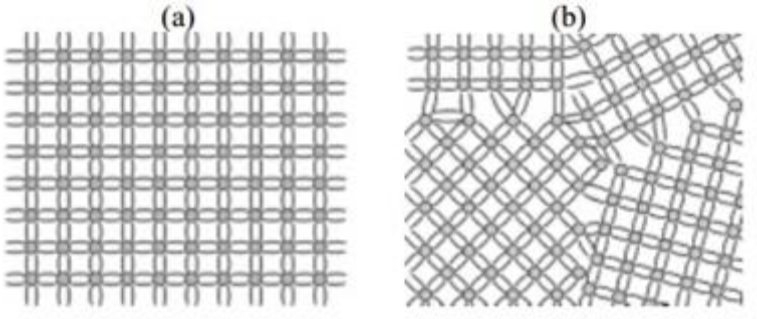


Figura 1.11 Estructuras monocristalina y policristalina

1.1.3.1.1. Primera generación

Las células de primera generación consisten en dispositivos de unión simple, de alta calidad y gran área. Las tecnologías de primera generación necesitan mucha energía y procesado manual, lo que impide una reducción significativa de los costes de producción.

Sus ventajas principales son una eficiencia máxima en la conversión en torno al 25% en el caso del silicio monocristalino y un 20% para el silicio multicristalino (cercano al límite teórico calculado por Shockley y Quessier en 1961). Las superficies de estas células son muy grandes, presentan una alta calidad y son fáciles de unir.

Desventajas principales: Parecen haber tocado techo en cuanto a la reducción de los costes de fabricación.

Los métodos de obtención de los materiales que componen esta generación de células solares serán definidos en los puntos (**1.1.3.3, 1.1.3.4 y 1.1.3.5)**

1.1.3.1.2. Segunda generación

En concreto, en este trabajo se tratarán las células denominadas de segunda generación, basadas en tecnologías de lámina delgada en colaboración con el Centro de investigaciones energéticas, medioambientales y tecnológicas (CIEMAT). Este tipo de células incorporan una lámina de semiconductor degenerado de tipo n que determina la cantidad de luz que entra dentro en el dispositivo y a su vez hace de electrodo frontal, y que se denomina óxido conductor transparente.

La principal razón por la que esta generación de células solares está empezando a ser relevante en el mercado es el importante ahorro de material semiconductor, a parte de tener costes de producción relativamente bajos.

En definitiva, gran parte de los estudios realizados en este tipo de células solares se basan en encontrar materiales semiconductores capaces de absorber luz solar en capas extremadamente finas, y que se depositen en grandes superficies sobre substratos baratos mediante métodos simples y de fácil adaptación a procesos industriales. Claramente dirigido, es decir, una estrategia claramente dirigida a la reducción de costes de producción.

En este punto, las técnicas de fabricación de este tipo de células tienen la ventaja de poder reducir la temperatura del proceso de forma

significativa y permiten emplear técnicas de producción en masa como el denominado *roll to roll*. Estas técnicas se caracterizan fundamentalmente por:

- El uso de substratos baratos y flexibles.
- La fabricación de módulos de forma continua a partir de un rollo de plástico flexible.
- La reducción del tiempo de producción, ya que se minimiza el número de procesos involucrados en su fabricación.
- El incremento de la reproducibilidad.
- El abaratamiento de costes de manufacturado y transporte.

En este tipo de dispositivos, la capa absorbente es la responsable de la fotogeneración de portadores y es la que da nombre a la célula. Destacan los siguientes semiconductores:

- Silicio amorfo y sus aleaciones con hidrógeno
- Telururo de cadmio (CdTe)
- Semiconductores de tipo CIS/CIGS (aleaciones de $Cu(In, Ga)(S, Se)_2$)

Figura 1.12 Célula solar de segunda generación

1.1.3.1.3. Tercera generación

Las tecnologías de tercera generación intentan mejorar la baja eficiencia eléctrica de los sistemas de segunda generación, mientras mantienen bajos costes de producción.

Estas células pueden superar el límite teórico de eficiencia de conversión solar para materiales de un solo umbral de energía, y la investigación actual se ha marcado un objetivo de eficiencia de conversión de entre el 30% y el 60%, manteniendo materiales y tecnologías de fabricación de bajo coste.

Existen unas pocas aproximaciones a estos increíbles ratios de conversión energética, entre los que se cuentan el uso de células fotovoltaicas **multi-unión**, la concentración del espectro incidente, el uso de generación termal por luz UV para mejorar la tensión o la recolección de portadores, o el uso del espectro UV para el funcionamiento nocturno.

1.1.3.2. Obtención del silicio con calidad FV para fabricación de células de primera generación.

Aunque el silicio sea muy abundante en la naturaleza, para obtener niveles de eficiencia en las células fotovoltaicas que sean eficientes, el material debe tener una pureza determinada. El primer paso consiste en obtener el denominado silicio de grado metalúrgico, cuya pureza está en torno al 99%. Esto se realiza en hornos de arco a 1800ºC. Se produce una reducción de cuarzo utilizando carbón, dando lugar al mencionado silicio y monóxido de carbón.

La producción anual y el coste se estiman aproximadamente en 1 millón de toneladas y 1€/Kg. De ellas, la industria microelectrónica emplea una pequeña fracción (aproximadamente el 3%), mientras que la mayor parte se emplea en metalurgia.

Con este porcentaje de pureza no es suficiente, por lo que el silicio de grado metalúrgico se hace reaccionar con ácido clorhídrico, obteniendo triclorosilano. Este pasa por columnas de destilación, donde se realizan procesos de purificación. Finalmente se hace reaccionar con hidrógeno, dando lugar a silicio de grado electrónico, obteniendo una pureza del 99.9999999%. Este proceso consume mucha energía y es costoso. Además, sólo el 40% del silicio obtenido es apto para aplicaciones de la industria microelectrónica. Este silicio

tiene la pureza deseada pero no la estructura cristalina necesaria, es por esto que se utilizan los métodos de obtención descritos en los puntos **(1.1.3.3 y 1.1.3.4).**

Tras la obtención de las piedras de **polisilicio** purificado el siguiente paso es realizar la cristalización del mismo, formando un conjunto de átomos totalmente ordenado (silicio cristalino), o un orden estructurado por segmentos (silicio policristalino)

1.1.3.3. Crecimiento por el método de la zona flotante

Este método de crecimiento se basa en una columna vertical de silicio policristalino sujeta en sus dos extremos por dos soportes. Una bobina por la que circula una corriente de radiofrecuencia rodea dicha columna y produce una fusión localizada del silicio en la sección de la bobina. La bobina empieza a fundir el silicio del extremo inferior, donde hay una semilla de silicio monocristalino. Al subir lentamente hace que la zona fundida del centro de la bobina se desplace hacia la parte superior. La zona fundida que queda debajo se recristaliza siguiendo una estructura monocristalina. La zona fundida queda por lo tanto flotante entre dos zonas sólidas. Debido a que en el silicio las impurezas en general y las metálicas en particular tienden a ir a la fase líquida, este proceso sirve además de purificación. Los lingotes obtenidos con este método poseen así menos impurezas que los crecidos con el método Czochralsky

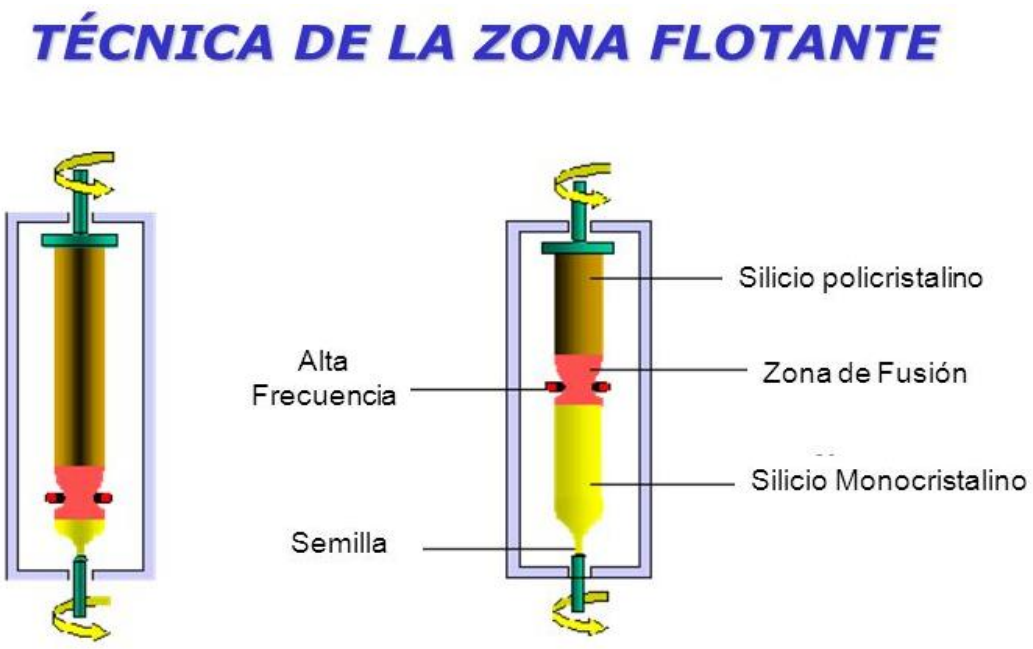


Figura 1.13 Cristalización por el método la zona flotante

Como el lingote no está en contacto con ningún crisol, el contenido en carbono y oxígeno es muy bajo. En cuanto a las impurezas metálicas, tenderán a acumularse en la parte superior del lingote, el cual no será rechazado. A mayor número de pasadas de la bobina de radiofrecuencia, menor será el contenido de impurezas del lingote.

1.1.3.4. Crecimiento por el método Czochralsky

En esta técnica, las piedras de polisilicio se funden en un crisol. La temperatura de este se controla para que esté justamente por encima del punto de fusión del silicio y no empiece a solidificarse. En el crisol se introduce una varilla que gira lentamente y tiene en su extremo un pequeño monocristal de silicio con la orientación cristalográfica deseada, que actúa como semilla. Al contacto con la superficie de semiconductor fundido, éste se agrega a la semilla, solidificándose con su rede cristalina orientada de la misma forma.

La varilla se va elevando y colgando de ella, se va formando un monocristal cilíndrico. El grosor del lingote dependerá del control de la temperatura y la velocidad de la varilla. Este procedimiento es el más utilizado en la actualidad para realizar el crecimiento de lingotes de silicio monocristalino.

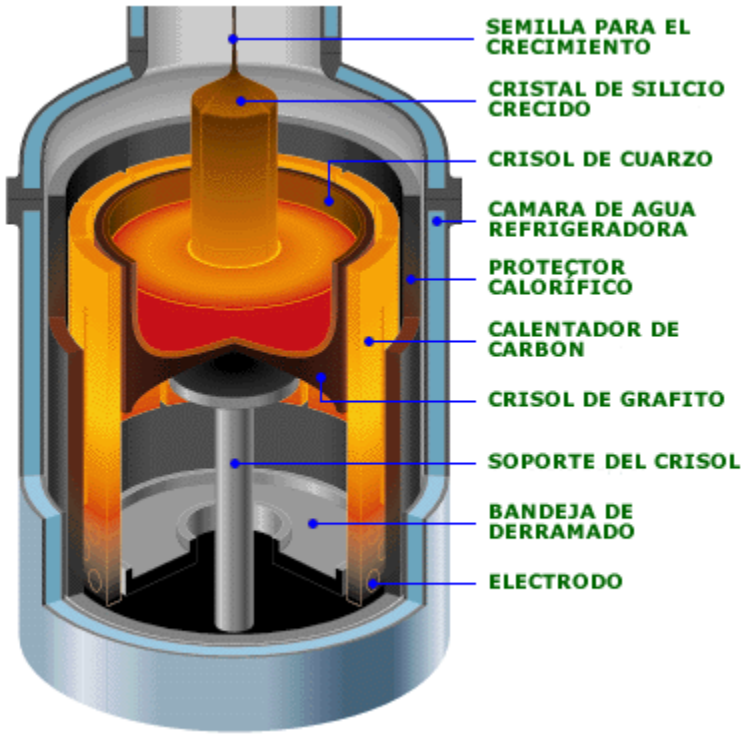


Figura 1.14 Cristalización por el método Czochralsky

1.1.3.5. Método de obtención del silicio multicristalino

El silicio multicristalino se obtiene mediante el proceso denominado **solidificación direccional** del polisilicio. Al solidificarse, el polisilicio, se forman múltiples cristales con orientaciones cristalográficas diferentes, de un tamaño en torno a uno o varios milímetros, formando así el silicio multicristalino (mc-Si). Este proceso de obtención consume menos energía que el método Czochralski, y por tanto, al reducir los costes de fabricación del material también se reducen los costes de fabricación de este tipo de células solares. De ahí que su participación en el mercado sea relevante.

Este proceso de obtención consume menos energía que el método Czochralski, al reducir los costes de fabricación del material se reducen los costes de estas células solares

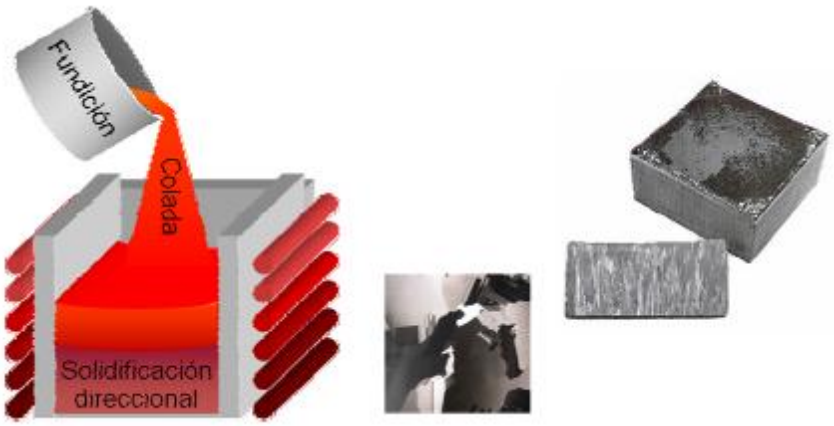


Figura 1.15 Cristalización del silicio multicristalino

1.1.3.6. Fabricación de células de lámina delgada

Las células solares de capa fina o capa delgada vienen fabricándose desde la década de los noventa. En ellas el semiconductor se deposita en forma de capa de espesor pequeño (del orden o inferior a la micra) sobre un sustrato de bajo coste (cristal o vidrio, en la mayoría de los casos). Los materiales que se suelen usar son el silicio amorfo, el cobre-indio-diselenio(CIS), el telururo de cadmio (CdTe) y el cobre-indio-galio-selenio (CIGS)

Las células de silicio amorfo poseen una estructura no cristalina. El silicio amorfo se puede depositar como una capa fina sobre diversos tipos de soportes, abriendo así la posibilidad de fabricar células fotovoltaicas flexibles. Su coste de fabricación es relativamente bajo, pues el silicio amorfo no requiere un proceso de cristalización previo, sino que basta extraer el oxígeno del silicio para obtenerlo. Sin embargo, uno de los problemas del silicio amorfo es su menor

eficiencia respecto del silicio monocristalino y policristalino. Esto es debido a que el silicio amorfo se degrada al poco tiempo de exposición a la luz solar, y su eficiencia se estabiliza en un valor inferior al inicial. El motivo de esto es la creación de una serie de defectos metaestables conocidos como efecto Staebler-Wronski, responsables de la reducción apreciable de su eficiencia durante las primeras semanas o incluso meses de operación. Este proceso de degradación puede limitarse en parte reduciendo el espesor de la lámina de a-Si:H, de forma que los portadores fotogenerados deban desplazarse distancias muy cortas hasta los electrodos.

Los materiales y aleaciones basados en el silicio amorfo constituyen una de las tecnologías más veteranas dentro de este campo y presentan, como ventajas más destacadas su abundancia y no toxicididad, la posibilidad de depositarlo en procesos a baja temperatura

En general, las temperaturas de fabricación de las células de segunda generación son más bajas y por tanto, la cantidad de energía consumida en el proceso es menor; además, su eficiencia es mayor en condiciones de luz difusa. No obstante, esta tecnología posee también desventajas, que en el caso del silicio amorfo es la menor eficiencia; y en los otros casos, el uso de cadmio, material sumamente tóxico, que hace el proceso de fabricación complejo y contaminante.

Los módulos de lámina delgada se fabrican en líneas con un elevado grado de automatización. No es posible producir en una primera fase las células, para encapsularlas posteriormente, tal y como ocurre con la tecnología de silicio cristalino. El crecimiento del semiconductor se realiza directamente sobre una lámina de vidrio, metal o plástico, que formará parte del módulo en su estado final. Esto es una de las ventajas más significativas de este tipo de células

En líneas generales, el proceso de fabricación del dispositivo en todas las tecnologías de lámina delgada consta de las siguientes partes:

- depósito del contacto frontal
- depósito de las capas semiconductoras
- depósito del contacto posterior.

En algunos casos el orden puede invertirse comenzando la fabricación del módulo por la cara posterior. Esto es habitual en todos los módulos de CIGS y en algunos módulos de silicio amorfo.

En la fabricación de los módulos de lámina delgada es clave conseguir una buena interfaz entre todas las capas depositadas, para evitar la formación de centros de recombinación y de tensiones internas que con el tiempo puedan dar lugar a diferentes mecanismos de degradación.

1.1.3.6.1. Depósito del contacto frontal

Como contacto frontal en los módulos de lámina delgada se utiliza un **óxido transparente conductor** (**TCO**) que se aplica de forma continua en toda la superficie del módulo. Este ha de tener una buena conductividad y una elevada transmisividad para dejar pasar la radiación incidente en el dispositivo. Además, ha de ser muy estable para resistir elevadas temperaturas en los siguientes pasos de fabricación del módulo. Entre los materiales más utilizados se encuentran el, SnO_2, el In_2O_3:SnO_2 (ITO), y finalmente el ZnO dopado con Aluminio, Galio, Indio o Boro.

En ocasiones se utilizan dos capas de TCO de modo que se aprovechan las distintas características de materiales diferentes. Por ello, el conocimiento de los parámetros ópticos de las distintas capas de TCO es muy importante.

El proceso más habitual para depositar el TCO es el **depósito físico en fase vapor** PVD (*physical vapour deposition*), generalmente empleando la técnica de **pulverización catódica** (*sputtering)*. En este caso no hay reacciones químicas, se trata de un proceso físico mediante el cual los átomos de un material se vaporizan tras ser bombardeados por iones muy energéticos de un gas inerte ionizado, que suele ser Argon. Estos iones que constituyen el denominado plasma son acelerados hacia el material que se quiere vaporizar mediante un campo eléctrico. Los átomos liberados tienden a condensarse de vuelta a su estado sólido al chocar con cualquier superficie en la cámara de pulverización, y en particular con el substrato sobre el que se pretende depositar el contacto.

1.1.3.6.2. Depósito de las capas semiconductoras: capa ventana y absorbedor.

Las células de lámina delgada suelen presentar una **heterounión**. Esto quiere decir que los materiales que forman la unión p-n son diferentes. La **capa ventana** es la primera capa de semiconductor a la que llegan los fotones. Para que la máxima cantidad de luz llegue a la

zona de la unión, esta capa ha de tener una energía del gap elevada, y además ser de muy bajo espesor. Esto también es importante para disminuir las pérdidas resistivas.

El **absorbedor** es el material semiconductor principal que forma la célula solar. Los principales materiales utilizados en los módulos de lámina delgada son el silicio amorfo hidrogenado, el CdTe y el CIGS.

Existen una gran variedad de técnicas para depositar los semiconductores en los módulos de lámina delgada. En los módulos de a-Si:H, es habitual utilizar un reactor de deposición química en fase vapor asistido por plasma (**PECVD)**, similar al utilizado para el depósito de capas de SiN en los módulos de silicio cristalino. Para depositar a-Si las temperaturas de proceso no han de ser muy elevadas, lo que permite utilizar substratos de bajo coste como plásticos, facilitando la fabricación de módulos flexibles. YA LO HAS DICHO ANTES

Controlando los gases precursores y los parámetros del reactor, se pueden ir depositando diferentes capas con la composición, dopado, y grosor deseadas, para crear múltiples tipos de configuración de células, con una o varias uniones. En lugar de una unión p-n, en la tecnología a-Si:H se utilizan uniones tipo ***p-i-n***: una capa de material tipo *p* con sólo unos 20 nm de espesor, seguida de una capa de material intrínseco (sin dopar) de mayor espesor, y otra capa muy fina de material tipo *n*. La ventaja de esta estructura es que se ensancha la zona del campo eléctrico que pasa a ocupar toda la zona *i*, en la cual la movilidad de los electrones y huecos es mucho mayor que en las zonas dopadas. De este modo los portadores generados en la capa *i* pueden ser arrastrados por el campo eléctrico y ser extraídos del dispositivo antes de recombinarse.

Los módulos de CdTe también utilizan CdS como capa ventana. En estos módulos, la técnica más común de depósito del CdS y el CdTe, se denomina **depósito por transporte de vapor** (*vapor transport deposition* o VTP). Se trata de una técnica muy rápida y que utiliza temperaturas medias. Consiste en la sublimación del material de partida, CdS o CdTe en forma de polvo, en una atmósfera inerte, seguida del transporte de los gases hacia el substrato, donde tiene lugar su solidificación y depósito. A continuación se suele realizar un paso de activación térmica, en presencia de $CdCl_2$ y O_2, que mejora algunas propiedades de los materiales, principalmente pasivando centros de recombinación.

Otro proceso alternativo es la aplicación de la capa CdS mediante **baño químico**, seguida por el **depósito electroquímico** de la capa CdTe (se genera un campo eléctrico entre el substrato y otro electrodo, introduciéndolo en una solución que contiene los materiales precursores que precipitan al ser atravesados por una corriente). También están en desarrollo otros procesos, como la aplicación de CdTe mediante serigrafía seguida de un paso térmico de activación.

1.1.3.6.3. Depósito del contacto posterior

Como contacto posterior se deposita una capa metálica que presente un buen contacto óhmico con el semiconductor. La técnica utilizada es generalmente pulverización catódica, como en el caso del TCO.

Es habitual añadir una capa adicional que actúa como **reflector**, de modo que la radiación no absorbida pueda ser devuelta al semiconductor incrementando así la eficiencia del dispositivo. Finalmente, cuando el metal elegido no es soldable, se deposita una capa adicional en la cara más externa del módulo, de algún material que permita la soldadura posterior de los terminales.

1.1.4. TEORÍA ESPECTRAL PARA LA CÉLULA SOLAR

La distribución espectral de la energía que proviene del sol se extiende entre las longitudes de onda de 250 nm a los 3 μm, aproximadamente. Esto, traducido a niveles se corresponde con 4,96 eV y 0,41 eV.

Teniendo en cuenta los factores que limitan el rendimiento de una célula solar, se establece un rango de valores de Eg que resultan adecuados a la hora de escoger los distintos materiales para fabricar las células solares, desde los 0,7 eV hasta los 2,2 eV.

Por definición, se denominan AM0 el espectro solar extra-atmosférico, y AM1.5 el espectro solar en la superficie de la tierra. En las referencias de las medidas de células solares se usa el espectro AM1.5: que se usa para las calibraciones, medidas de rendimiento, comparaciones y caracterizaciones, y tiene un valor normalizado de **irradiancia** de 1000 Wm-2(denominado 1 sol). El espectro AM0, se usa para las aplicaciones espaciales, que no deben tener en cuenta la

absorción atmosférica, y corresponde a una irradiancia de 1366 Wm-2(denominado constante solar).

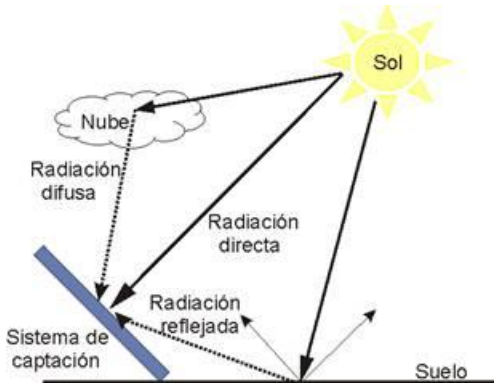


Figura 1.16 Radiación incidente en módulos

Los fotones que llegan a la superficie del semiconductor pueden ser absorbidos, reflejados o transmitidos. Los fotones reflejados o transmitidos no contribuirán a la conversión de energía. En el sistema de captación, o módulo (conjunto de células conexionadas entre sí) no solamente contribuye a la generación de electricidad la radiación directa, como vemos en la (figura 1.16) la radiación difusa y la radiación reflejada, inciden sobre el sistema de captación, aumentando así la radiación total que capta el módulo. En relación con los fotones absorbidos tenemos tres casos:

- Eph<Eg La energía del fotón es menor que la del ancho de banda prohibida del semiconductor, por lo que no intervendrá a la generación de corriente. El fotón no es absorbido por el material

- Eph=Eg el fotón posee la energía mínima para producir un par electrón-hueco

- Eph>Eg el fotón tiene mucha energía y provoca que un electrón pase a la banda de conducción con energía superior a la del estado de mínima energía en esa banda. La diferencia entra esta energía inicial y la del estado de mínima energía disponible en la banda de conducción se disipa en forma de calor.

Tal y como se ha indicado en apartados anteriores, en las células solares de lámina delgada en las que la luz incide a través del substrato, el número de fotones que entra en el dispositivo y que se emplea para la generación de electricidad, viene determinado por la

cantidad de luz que la estructura substrato y capa de óxido conductor transparente (OCT) dejen pasar. Para sea máxima la cantidad de luz que llega a la zona de la unión, esta capa de OCT tiene que ser transparente en la parte del espectro en la que responde la zona activa de la célula, y además ser conductora para disminuir en la medida de lo posible las pérdidas eléctricas del dispositivo.

Por ello, es importante conocer bien sus propiedades ópticas. De ahí la necesidad de contar con las herramientas adecuadas, como la que se ha desarrollado en este proyecto.

1.1.4.1. Leyes básicas de interacción radiación-materia. Transmisión, absorción y reflexión.

Cuando un rayo de luz se propaga por un medio y alcanza el límite que lo separa de un segundo medio, puede suceder que retorne al primero (reflexión), o que lo atraviese y que ingrese al segundo medio donde parte se convertirá en otra forma de energía (absorción) y parte no cambiará (transmisión). Dos, o los tres de dichos fenómenos ocurren simultáneamente, y como la energía no se puede destruir, la suma de la energía transmitida, absorbida y reflejada debe ser igual a la energía incidente.

1.1.4.1.1. Reflexión

Cuando unas ondas de cualquier tipo inciden sobre una barrera plana como un espejo, se generan nuevas ondas que se mueven alejándose de la barrera. Este fenómeno se denomina reflexión.

En el caso de la luz, cuando esta luz es reflejada por una superficie, un porcentaje de dicha luz se pierde debido al fenómeno de absorción. La relación entre la luz reflejada y la luz incidente se denomina **reflectancia** de la superficie.

La cantidad de luz que refleja una superficie dada y la forma en que dicha luz es reflejada se determina por las propiedades de reflexión de la superficie.

La reflexión especular se produce cuando la superficie reflectora es lisa, entendiendo como tal, una superficie con una rugosidad muy inferior a la longitud de onda de la radiación incidente. Dicha reflexión obedece a dos leyes fundamentales:

- El rayo incidente, el rayo reflejado y el normal a la superficie en un punto de incidencia se trazan en un mismo plano.

- El ángulo de incidencia (i) es igual al ángulo de reflexión (r).

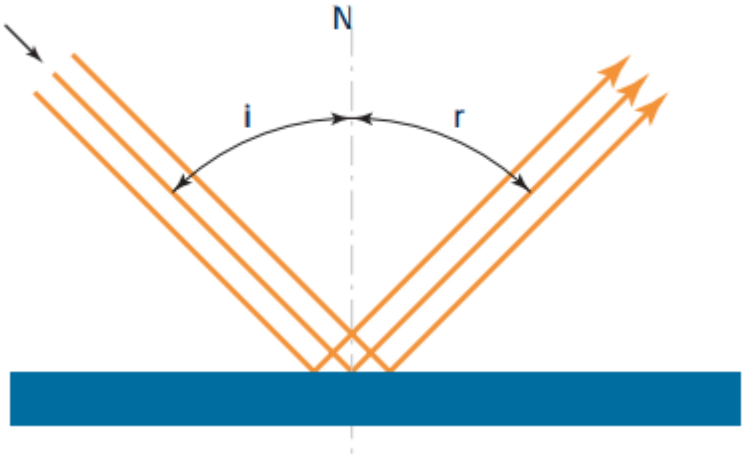


Figura 1.17 Reflexión especular.

1.1.4.1.2. Transmisión

Es el paso de una radiación a través de un medio sin cambio de frecuencia de las radiaciones monocromáticas que la componen.

La relación entre la luz transmitida y la luz incidente se denomina **transmitancia** del material.

En la transmisión, el haz que incide sobre un medio, la atraviesa y sale de él sin modificar su longitud de onda. Los medios que cumplen esta propiedad, se les denomina cuerpos “transparentes” y permiten ver con nitidez los objetos colocados detrás de ellos.

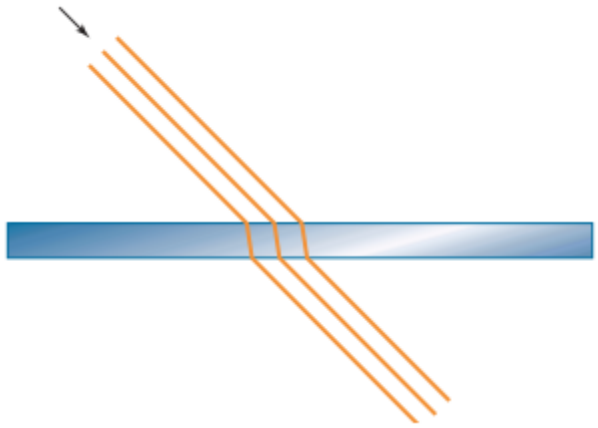


Figura 1.18 Transmisión

1.1.4.1.3. Absorción

Se denomina absorción a la transformación de la energía radiante en otra forma de energía, generalmente en forma de calor. Este fenómeno es una característica de todas las superficies que no son completamente reflectoras, y de los materiales que no son totalmente transparentes, para la longitud de onda estudiada.

La relación entre la luz absorbida y la luz incidente se denomina **absortancia** del material. La absorción de ciertas longitudes de onda de luz se denomina absorción selectiva. En general, los objetos de color le deben su color a la absorción selectiva.

De este modo, la potencia de luz transmitida, $P_{o,}$ por un material de anchura, L, se relaciona con la potencia de entrada sobre la misma, $P_{i,}$, según la siguiente expresión:

$P_o = P_i . e^{-\alpha L}$

, donde α es el coeficiente de absorción del material. En la ecuación anterior cada una de las variables pueden ser dadas en función de la longitud de onda, λ, o bien se puede eliminar la dependencia en λ si se integran a todo el rango de longitudes de onda del espectro analizado.

1.1.4.2. Modelos matemáticos.

El proceso de caracterizar las propiedades ópticas de capas ventana tiene dos pasos, por un lado, la realización de la medida experimental (en este caso, la medida de transmisión), y por otro, el ajuste de los parámetros del modelo óptico de la capa de manera que la transmisión obtenida mediante las simulaciones se aproxime lo más posible a las medidas experimentales. En este proyecto se han modelado las propiedades ópticas de la capa a través de su índice de refracción y su coeficiente de absorción. Los modelos empleados para estas propiedades ópticas son los siguientes:

1.1.4.2.1. Índice de refracción. Ecuación de Sellmeier

En óptica, la ecuación de Sellmeier es una relación empírica entre el índice de refracción "n" y la longitud de onda "λ", para un medio transparente particular. La forma habitual de la ecuación para cristales es:

$$n^2(\lambda) = 1 + \frac{B_1\lambda^2}{\lambda^2 - C_1} + \frac{B_2\lambda^2}{\lambda^2 - C_2} + \frac{B_3\lambda^2}{\lambda^2 - C_3},$$

Donde B1, B2, B3 y C1, C2, C3 son los coeficientes de Sellmeier determinados experimentalmente.

Hay que tener en cuenta que esta λ es la longitud de onda en el vacío, no en el material en el que medimos.

Esta ecuación se utiliza para determinar la dispersión de la luz en un medio refractivo. La ecuación fue deducida en 1871 por W. Sellmeier, a partir del desarrollo del trabajo de Augustin Cauchy en la ecuación de Cauchy para modelos de dispersión.

En el caso del presente trabajo, se ha empleado una aproximación a la ecuación de Sellmeier en la que se considera sólo una longitud de onda de resonancia, λ_0, que se suele situar cerca del gap del semiconductor:

$$n^2 = P + \frac{A\lambda^2}{\lambda^2 - \lambda_0^2}$$

Los parámetros P, A y λ_0, son los parámetros de ajuste.

1.1.4.2.2. Coeficiente de absorción, función sigmoidal

Muchos procesos naturales y curvas de aprendizaje de sistemas complejos muestran una progresión temporal desde unos niveles bajos al inicio, hasta acercarse a un clímax transcurrido un cierto tiempo; la transición se produce en una región caracterizada por una fuerte aceleración intermedia.

La función sigmoidal permite describir esta evolución. Su gráfica tiene una típica forma de "S".

A menudo la función sigmoidal se refiere al caso particular de la función logística, cuya gráfica se muestra en la **(figura 1.19)** y que viene definida por la siguiente fórmula:

$$P(t) = \frac{1}{1 + e^{-t}}$$

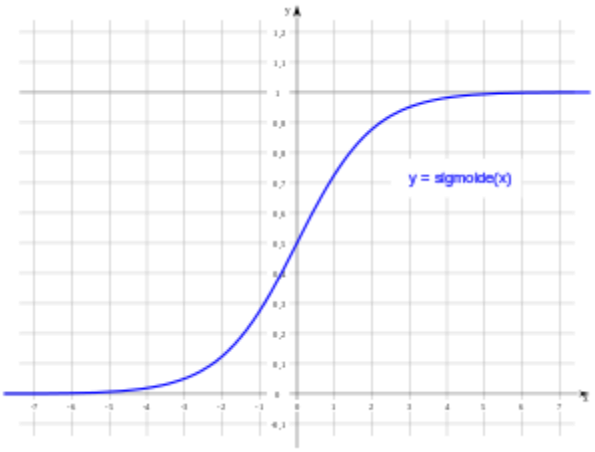


Figura 1.19 Función sigmoide

Esta función se ha empleado para la descripción semiempírica del comportamiento del coeficiente de absorción de los semiconductores de gap directo, empleando la siguiente aproximación:

$$\alpha = \frac{\alpha_0}{1 + e^{\frac{E_g - E}{\Delta E}}}$$

, donde α_0, E_g y ΔE son parámetros de ajuste de la ecuación y representan el valor del coeficiente de absorción a energías muy superiores a las de gap, la energía de gap y la anchura del borde de absorción (donde se produce cambio en la absorción desde 0 hasta la máxima), respectivamente.

1.1.4.2.3. Urbach tail

El gap de energía en los semiconductores se ve afectado por la presencia de colas de bandas que tienen distintos orígenes. Estas colas afectan a la respuesta óptica y eléctrica del material. La absorción óptica en una gran variedad de semiconductores, presenta un crecimiento exponencial con la energía fotónica ($\hbar\omega$) en la región del borde de absorción justo debajo de la brecha de energía Este crecimiento exponencial fue observado por Urbach en 1953 y es conocido como colas de bandas, colas de Urbach (Urbach tails). A una temperatura dada este crecimiento, tiene la siguiente forma:

$$\alpha(\hbar\omega, T) = \alpha_0 \exp\left[-\sigma(\hbar\omega_0 - \hbar\omega) / kT\right]$$

, donde $\boldsymbol{\alpha_0}$ es la absorción óptica cuando $\boldsymbol{\hbar\omega = \hbar\omega_0}$, $\boldsymbol{\hbar\omega_0}$ es una energía llamada de convergencia, σ el parámetro de "*steepness*" y ***k*** la constante de Boltzmann. Esto significa que el gráfico de ln $\boldsymbol{\alpha}$ vs $\boldsymbol{\hbar\omega}$, cerca del borde de banda, puede ser representado por una línea recta. Hay ciertos semiconductores en los que esta cola es difícil de observar en las curvas de absorción porque está solapada por la presencia de otros defectos cerca del borde de banda o con las oscilaciones producidas por interferencias durante la medida.

Hay un acuerdo general para los semiconductores cristalinos, y es que el ancho de la cola exponencial o energía de Urbach es una medida directa del desorden inducido por la temperatura y que incluye además el desorden estructural producto de defectos estructurales, dislocaciones, tensiones, etc. La energía de Urbach viene representada por la relación:

$$E_u = kT / \sigma$$

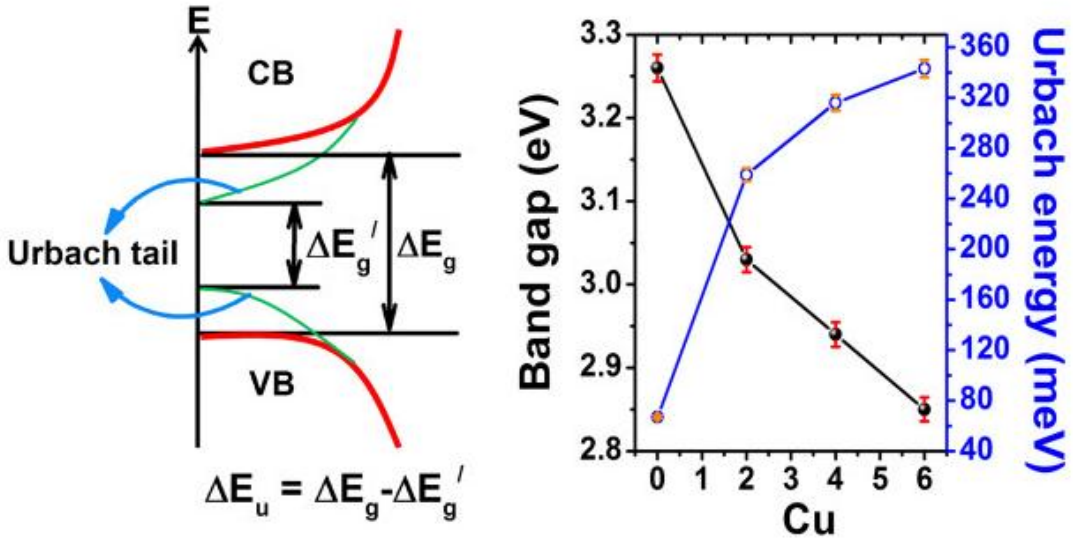


Figura 1.20 Cola de bandas y relación entre energía de Urbach y gap

En la representación de la izquierda de la (**figura 1.20**), vemos de manera genérica cómo, las *Urbach tails* o colas de banda que se generan entre las bandas de los semiconductores, provocan una disminución de la energía de gap. La brecha de energía en los

semiconductores (energía de gap) es afectada por la presencia de estas colas de banda, que tienen distintos orígenes, como pueden ser:

- Interacción portador-portador,
- Interacción portador-impureza,
- Interacción impureza-impureza,
- Interacción electrón-fotón.
- Defectos de la red cristalina

En la imagen de la derecha de la **(figura 1.15),** se representa, con datos numéricos, este efecto para el caso concreto del cobre. Se puede observar cómo a medida que la energía de la cola de banda aumenta, produce una disminución de la misma magnitud en la energía del *gap* .

En el punto inicial la energía del gap es de 3.25eV y la energía de Urbach de 0.065eV, cuando la energía de Urbach aumenta en unos 0.3eV, alcanzando los 0.35eV, la energía del gap disminuye de la misma manera, hasta llegar a 2.85eV, cumpliendo así la ecuación:

$\Delta Eu = \Delta Eg - \Delta Eg'$

, donde Eu representa a la energía de Urbach, Eg del semiconductor es la energía de gap y Eg' es la energía de gap resultante, teniendo en cuenta la energía de Urbach.

1.1.5. ¿Por qué es interesante el estudio?

En este trabajo se tratarán las células denominadas de segunda generación, basadas en tecnologías de lámina delgada. Los materiales empleados en dichas células se desarrollaron para abordar los problemas de requerimientos de energía y costes de producción de las células solares.

Las técnicas alternativas de producción como la deposición por solución, deposición por vapor, *electro-plating*, y el uso de inyectores ultrasónicos, representan una ventaja, ya que disminuyen significativamente el procesado a alta temperatura.

Es de señalar que está aceptado que conforme las técnicas de producción vayan mejorando, los costes de producción llegarán a estar dominados por el coste de los materiales constituyentes, tanto el substrato de silicio como de la cubierta de vidrio.

En la actualidad, la eficiencia eléctrica de esta segunda generación está entre el 5% y el 12%, con un empuje claro en los próximos años hasta el 10% - 15%. En el futuro, se espera que estas tecnologías puedan superar el 20%. Aunque en términos de eficiencia, las células de segunda generación no alcanzan a las células basadas en tecnología de silicio monocristalino, como vemos en la **(figura 1.21)**

		Tecnología	Eficiencia record	Eficiencia típica	Cuota de mercado	Ventajas	Desventajas
Tecnologías comerciales	Silicio cristalino	Mono	25,0% / 21,4%	14-16%	33% (9,1 GW)	Altas eficiencias, tecnología madura, amplia experiencia	Costes elevados
		Multi	20,4% / 17,5%	13-15%	53%		
	Lámina delgada	a-Si	11,9%	5-9%	5,0%	Proceso económico, material no tóxico y abundante	Eficiencia baja, degradación Staebler-Wronski
		CdTe	16,7%	11%	5,3%	Eficiencia media, proceso maduro y bajo coste	Toxicidad del Cd. Escasez del Te
		CIGS	19,6%	10-12,5%	1,6%	Buenas eficiencias	Proceso complejo y caro

Figura 1.21 Comparativa entre generación I y II de células solares

Los principales fabricantes están tendiendo a estas tecnologías de segunda generación, la comercialización de estas nuevas células se está mostrando lenta. En el 2007, la producción de CdTe representaba el 8.9% del mercado total, el *thin-film* de silicio el 5,2%, y el CIGS el 0,5%.

Estas tecnologías prometen en el futuro mayores eficiencias de conversión y reducciones altamente significativas en los costes de producción.

Otra de las ventajas que presenta la tecnología de lámina delgada es que el crecimiento de los distintos materiales semiconductores se realiza directamente sobre el substrato, ya sea vidrio, metal o plástico, que formará parte del módulo en su estado final. En la fabricación de los módulos de lámina delgada es clave conseguir una buena interfaz entre todas las capas depositadas, para evitar la formación de centros de recombinación y de tensiones internas que con el tiempo puedan dar lugar a diferentes mecanismos de degradación.

Recientemente se ha planteado el desarrollo de nuevas capas transparentes que permita sustituir el ITO por materiales menos costosos y más estables, todo ello manteniendo una elevada transparencia en el rango de longitudes de onda de interés para las células solares basadas en silicio (entre 400 nm y 1100 nm).

Por lo tanto, las previsiones de mercado, el aumento de las eficiencias, la disminución de los costes de fabricación y la posibilidad de realizar el crecimiento de los semiconductores directamente sobre el substrato hacen muy interesante realizar nuevas investigaciones en tecnologías de lámina delgada.

1.1.6. Objetivos del proyecto

En este trabajo se propone el desarrollo de una herramienta basada en Matlab para el análisis de las medidas ópticas de capas ventana. Dichas medidas se realizarán con un espectrofotómetro comercial.

A través de esta aplicación se estimarán las propiedades ópticas de las capas (coeficiente de absorción e índice de refracción), lo que permitirá establecer cuál de las composiciones del compuesto y de las condiciones ‘de su depósito promete una mejor utilidad en el campo de aplicación.

En concreto, la herramienta se aplicará para comprobar su funcionamiento a capas basadas en aleaciones de Zn-In-Al-Sn que se están desarrollando actualmente en el CIEMAT. En este punto cabe señalar que el depósito del material queda fuera del alcance de este trabajo. Las características obtenidas con la herramienta servirán de ayuda para elegir las capas más apropiadas para aumentar la eficiencia de las células solares en desarrollo.

La caracterización óptica de capas transparentes se realiza fundamentalmente midiendo el espectro de transmisión de la capa. A partir de este espectro y suponiendo unas relaciones para los parámetros ópticos preestablecidas para este tipo de materiales, se pueden obtener las propiedades ópticas de las capas en función de la longitud de onda. Ahora bien, la obtención de los parámetros ópticos que determinan las propiedades de las capas exige un proceso de ajuste por mínimos cuadrados de los datos experimentales a una función prueba. Este proceso suele ser tedioso y requiere de potentes algoritmos de ajuste a funciones de varios parámetros.

Es por tanto de inmediata aplicación el desarrollo de una herramienta para la realización de estos ajustes con una apropiada interfaz de usuario, de forma que se pueda utilizar en entornos de investigación y desarrollo sin necesidad de conocimientos de algoritmos de ajuste y programación. El entorno Matlab reúne los requisitos para el desarrollo de dicha herramienta, ya que ofrece potentes librerías de ajuste que permiten la definición de parámetros y métodos, así como una interfaz de usuario (GUI) para su uso eficiente y rápido.

Por lo tanto, en este trabajo se desarrollará dicha herramienta teniendo en cuenta los modelos ópticos más comúnmente utilizados en este tipo de materiales, sus parámetros más importantes así como una interfaz de usuario para realizar las tareas de ajuste, introducción de parámetros de partida, selección del modelo a emplear, integración de nuevos modelos y manejos de ficheros y resultados para su posterior análisis

.

1.2. DESCRIPCIÓN EXPERIMENTAL

Esta sección expone el desarrollo del proyecto con una introducción a la toolbox GUIDE de Matlab, para lograr un mayor entendimiento de la labor realizada.

En lo que respecta a los modelos matemáticos para el ajuste de a los datos experimentales, aunque se han implementado las funciones clásicas, el proyecto no se ha particularizado para un modelo concreto, sino que permite la implementación de nuevos modelos, orientando a si la interfaz a dos posibles usos. Por un lado un usuario con conocimientos básicos sobre los distintos modelos con los que se trabaja en la caracterización óptica de láminas delgadas que sólo pretende usar las funciones disponibles, para obtener unos datos y resultados sin modificar parámetros no visibles, y por otro, un usuario experto que podrá acceder prácticamente a todos los parámetros disponibles, utilizando así los modelos, que según sus conocimientos mejor se puedan adaptar al sistema que se esté sometiendo a estudio, obteniendo unos resultados más fiables en la caracterización desde la GUI.

Para facilitar la utilización de la interfaz a los usuarios, se pensó desde un principio en un esquema que incorporase diversas ventanas, dedicada cada una de ellas a una parte especifica del proceso, de manera que la propia interacción con el programa guiase al usuario en el proceso.

Este proyecto también ha consistido en el estudio teórico de ecuaciones y modelos sencillos empleados comúnmente para modelar los parámetros ópticos lineales de los materiales semiconductores. La parte práctica ha consistido en el estudio de la programación en MATLAB y GUIDE.

1.2.1. MATLAB como entorno de desarrollo

MATLAB es un entorno de cálculo técnico de altas prestaciones para cálculo numérico y visualización, con las siguientes aplicaciones principales:

- Análisis numérico
- Cálculo matricial

- Procesamiento de señales
- Gráficos

Todo ello se integra en un entorno fácil de usar, donde los problemas y las soluciones son expresados como se escriben matemáticamente, sin la programación tradicional. El nombre *MATLAB* proviene de "*MATrix LABoratory*" (Laboratorio de Matrices).

MATLAB fue escrito originalmente para proporcionar un acceso sencillo al software matricial desarrollado por los proyectos *LINPACK* y *EISPACK*, que juntos representan lo más avanzado en programas de cálculo matricial. *MATLAB* es un sistema interactivo cuyo elemento básico de datos es una matriz que no requiere dimensionamiento. Lo que permite resolver muchos problemas numéricos en una fracción del tiempo que llevaría hacerlo en lenguajes como *C*, *BASIC* o *FORTRAN*.

MATLAB ha evolucionado en los últimos años a partir de la colaboración de muchos usuarios. En entornos universitarios se ha convertido en la herramienta de enseñanza estándar para cursos de introducción al álgebra lineal aplicada, así como cursos avanzados en otras áreas. En la industria, *MATLAB* se utiliza para investigación y para resolver problemas prácticos de ingeniería y matemáticas, con un gran énfasis en aplicaciones de control y procesamiento de señales. *MATLAB* también proporciona una serie de soluciones específicas denominadas *TOOLBOXES*.

Probablemente la característica más importante de *MATLAB* es su capacidad de crecimiento. Esto permite convertir al usuario en un autor contribuyente, creando sus propias aplicaciones. En resumen, las prestaciones más importantes de *MATLAB* son:

- Escritura del programa en lenguaje matemático.
- Implementación de las matrices como elemento básico del lenguaje, lo que permite una gran reducción del código, al no necesitar implementar el cálculo matricial.
- Implementación de aritmética compleja.
- Un gran contenido de órdenes específicas, agrupadas en *TOOLBOXES*.
- Posibilidad de ampliar y adaptar el lenguaje, mediante ficheros de *script* y funciones .m.

El sistema de desarrollo de MATLAB trabaja interpretando las órdenes dadas al sistema, tanto desde la consola interactiva, como a través de ficheros de texto. Este sistema no diferencia entre rutinas internas de la aplicación y rutinas creadas por el usuario – todas las rutinas y funciones internas de la aplicación son ficheros que se pueden modificar o directamente sustituir por una función propia del programador.

Todas las funciones se deben declarar en el fichero abierto en el momento, o acceder a través de un fichero con el mismo nombre. Esto permite ir creando funciones y saltos del programa, a través de ficheros.

El sistema de programación no permite estructuras tipo "*go to*", sino que todo se ejecuta en una pasada de programa, siendo la programación la que debe esperar la entrada del usuario o reconducir los procedimientos en función de los intereses del programa.

Además de los sistemas de programación descritos, MATLAB incorpora un sistema de creación de aplicaciones rápida, basado en eventos y formularios GUI, similar a lo que han sido hasta la fechas los entornos "visuales". Este sistema, denominado GUIDE, permite la creación gráfica de entornos de formularios, construyendo la estructura de programación que los dibuja en la pantalla, y permitiendo activar una función en función del evento generado.

Por ejemplo, se puede dibujar un botón en la pantalla, y especificar que cuando el usuario haga "*click*" en el botón, se active la función "*botón_click_1*", que se programará en MATLAB para que actúe en consecuencia.

Este entorno permite, como comentamos, la creación de interfaces gráficas de usuario, con cierta comodidad, programando únicamente las llamadas a los eventos producidos, mediante *callbacks*.

Con la realización de un interfaz gráfico de usuario, es posible modificar cualquier parámetro del sistema creado. Con este GUI, no solo se modifican los parámetros sino que además se puede guardar los datos y los cambios en el sistema, habilitando de este modo la posterior consulta y representación de los resultados obtenidos con anterioridad.

1.2.2. Introducción a GUIDE

GUI (*Graphical User Interface*, en español Interfaz Gráfica de Usuario), es un entorno de trabajo para la programación visual que nos proporciona Matlab para elaborar y ejecutar programas que necesiten ingreso continuo de datos. Tiene la propiedad básica de todos los programas visuales como Visual Basic o Visual C++.

Es una herramienta de trabajo que se extiende en el soporte de Matlab, planificada para crear interfaces gráficas para el usuario fácil y rápidamente, dando respaldo al diseño y presentación de los elementos de control de la interfaz, disminuyendo el esfuerzo al nivel de seleccionar, utilizar y personalizar propiedades.

Una vez que los elementos están colocados de una manera visual aceptada en el GUI del archivo (*.fig), se editan las funciones de llamada (*callback*) de cada uno de los elementos en el archivo GUI (*.m). El código que introducimos en el (*.m), se ejecutará cuando el elemento sea utilizado.

GUI está diseñado para tener que ofrecer menos esfuerzo en el proceso de aplicación de la interfaz gráfica. En el diseño de una GUI es muy importante el editor de propiedades (*property editor*), sus componentes se encuentran disponibles en cualquier momento que se esté trabajando con los controles de Matlab. El editor de propiedades se puede concebir como una herramienta de trazado y asistente de codificación (revisión de nombres y valores de propiedades). Cuando se fusiona con el panel de control, el editor de menú, y herramienta de alineación, resulta el control de los gráficos en Matlab.

Para llegar a entender a un nivel básico la forma de programar una interfaz, solo se necesita entender cinco comandos: uimenu, uicontrol, get, set y axes. No obstante, lo que hace verdaderamente refinados a estos comandos es el gran número de maneras de uso.

La interfaz gráfica está creada por dos archivos uno (*.m) y otro (*.fig).

Cada vez que se adicione un nuevo elemento en la interfaz gráfica, se genera automáticamente código en el archivo (*.m).

En el archivo (*.m), se encuentra un procesador de texto donde se programan todas las funciones que hacen referencia a todos los objetos

presentado en el archivo (*.fig). Las funciones creadas por el comando "*function*", hacen referencia al objeto mencionado por la denominación del nombre, pudiendo así modificar sus propiedades recogidas en el (*property inspector*) de manera interactiva.

Todos los valores de las propiedades de los elementos (color, valor, posición, string…) y los valores de las variables transitorias del programa se almacenan en una estructura, los cuales son accedidos mediante un único y mismo identificador para todos éstos. Ejemplo:

```
[Nombre-variable] = 1;
handles.[Nombre-variablesalida] = [Nombre-variable];
guidata (hObject, handles);
```

En este caso, vemos que el valor "1" queda grabado y puede ser llamado a través de la variable. El valor queda memorizado gracias al manejador o también llamado identificador. *Handles* es, en este caso, el identificador de los datos de la aplicación. Por otro lado las variables no quedarían memorizadas sin el comando:

```
guidata (hObject, handles);
```

Guidata, es la sentencia para salvar los datos de la aplicación, es la función que guarda las variables y propiedades de los elementos en la estructura de datos de la aplicación, por lo tanto, como regla general, en cada subrutina se debe escribir en la última línea. Esta sentencia garantiza que cualquier cambio o asignación de propiedades o variables quede almacenado.

1.2.2.1. ¿Cómo se arranca un GUI en Matlab?

Es posible iniciar un nuevo proyecto de dos maneras:

- Ejecutando la instrucción `guide` en la ventana de comandos:
- Haciendo un clic en el icono que muestra la **(figura 2.1):**

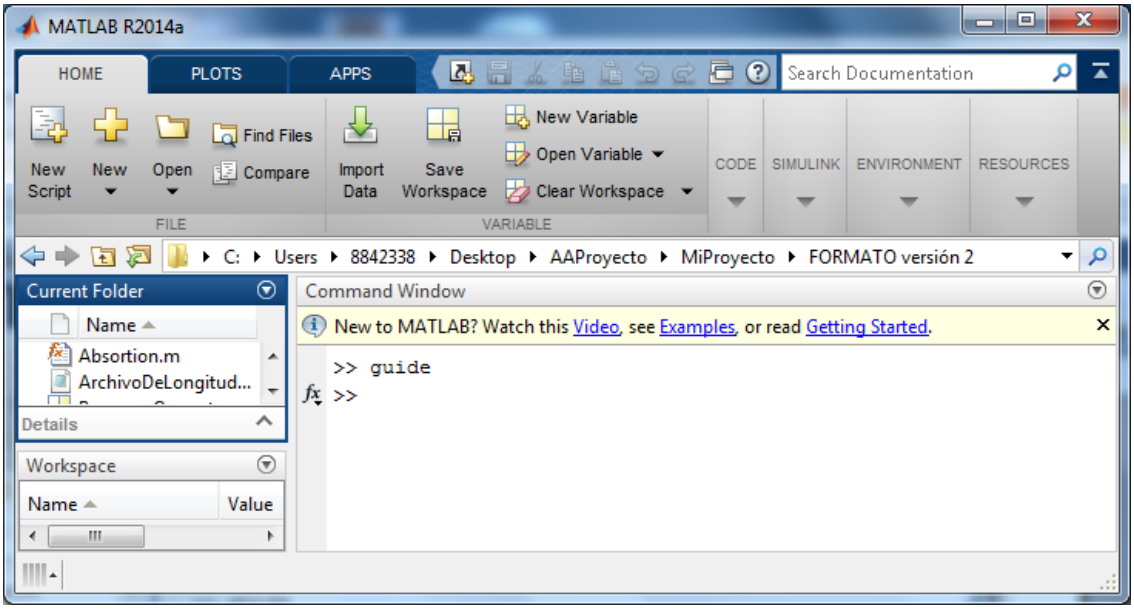


Figura 2.1 Iniciar GUI desde la consola.

Al ejecutarlo se presenta el siguiente cuadro de diálogo que muestra la **(figura 2.2)**

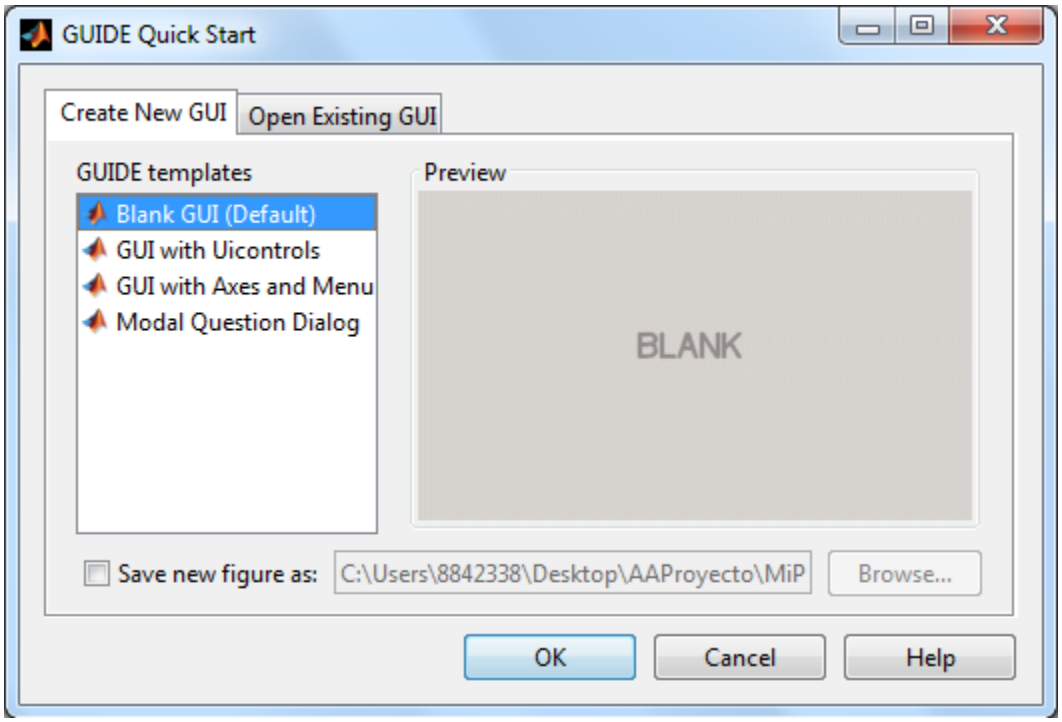


Figura 2.2 Ventana de inicio GUI.

En esta ventana las posibilidades de actuación son las siguientes:

- *Blank GUI* (*Default*): La opción de interfaz gráfica de usuario en blanco (opción predeterminada), presenta un formulario nuevo, en el cual se puede diseñar el nuevo programa.

- *Blank GUI* (*Default*): La opción de interfaz gráfica de usuario en blanco (viene predeterminada), nos presenta un formulario nuevo, en el cual podemos diseñar nuestro programa.

- *GUI with Uicontrols*: Esta opción presenta un ejemplo en el cual se calcula la masa, dada la densidad y el volumen, en alguno de los dos sistemas de unidades. Se puede ejecutar este ejemplo y obtener resultados, es un ejemplo muy básico pero interesante para conocer el funcionamiento de los bloques "*Edit Text*" .

- *GUI with Axes and Menú*: Esta opción es otro ejemplo, el cual contiene el menú File con las opciones *open*, *print* y *close*. En el formulario tiene un *Pop-up menú*, un *push button* y un *objeto axes*.

- *Modal Question Dialog*: Con esta opción se muestra en la pantalla un cuadro de diálogo común, el cual consta de una pequeña imagen, una etiqueta y dos botones "*Yes*" y "*No*", dependiendo del botón que se presione, el GUI retorna el texto seleccionado.

En el caso de este proyecto se ha elegido la primera opción, *Blank GUI*, y tenemos la ventana que muestra la **(figura 2.3)**

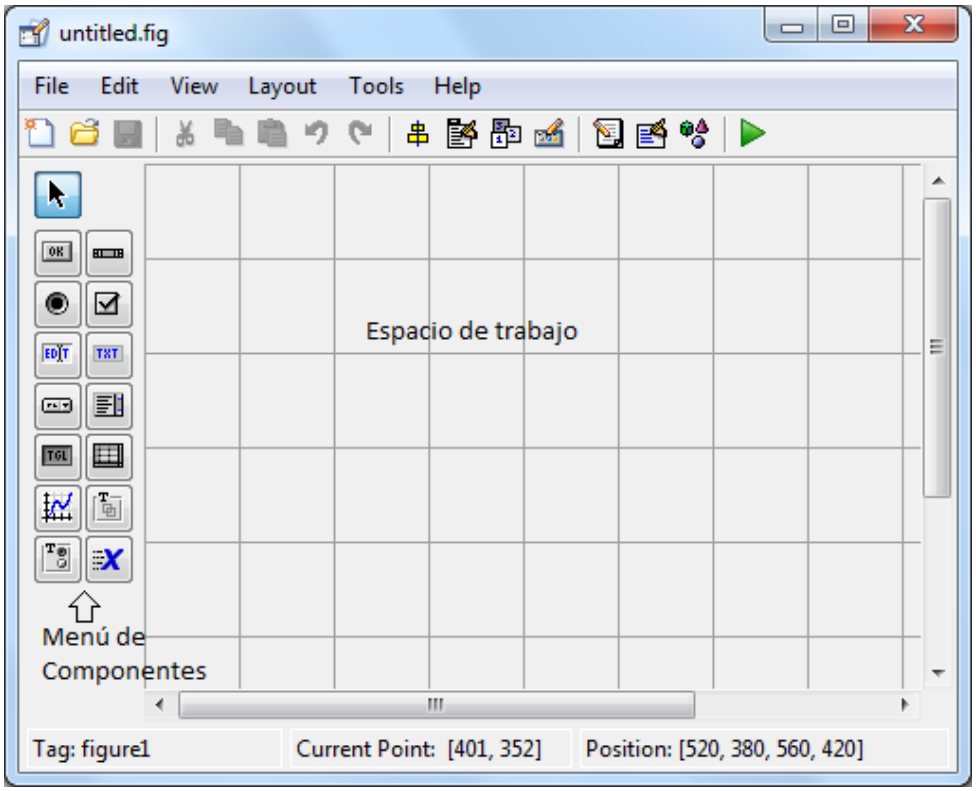


Figura 2.3 Entorno de diseño GUI

La interfaz gráfica cuenta con las siguientes herramientas:

 Alineación de Componentes (*Alignment tool*): esta opción permite alinear los componentes que se encuentra en el área de trabajo (*Layout Área*) de manera personalizada.

 Editor de menú (*Menú Editor*): El redactor de Menú crea menús de ventana y menús de contexto.

 Editor de orden de etiqueta, enumera y ordena los elementos

 Editor del M-file, presenta el archivo donde se escribe el código

 Propiedades de objetos (*Property Inspector*): Con esta opción se asignan y modifican las propiedades de cada objeto en forma personalizada.

 Navegador de objetos. Navegador de Objetos (*Object Browser*): Muestra todos los objetos que se encuentra en la figura

1.2.2.2. Callback

Matlab Guide utiliza el método de programación por retrollamada o "*callback*", este tipo de funciones permiten la ejecución de fragmentos de código (subrutinas) mediante su activación dentro de otra función, al utilizar el elemento asociado a ese "*callback*".

Una vez haya terminado la ejecución de la subrutina se devolverá la ejecución del programa al punto donde se llamó a la misma. Esta aproximación se presenta como una forma muy sencilla y útil en el desarrollo de interfaces gráficas con una gran cantidad de elementos. Además de ayudar a la estructuración del código.

1.2.3. REALIZACIÓN DE LA INTERFAZ

En esta sección se tratará de aclarar de que forma se han utilizado los diversos elementos que GUIDE proporciona, si bien es cierto que cada elemento ofrece una infinidad de posibilidades y en esta interfaz han sido utilizados de diversas formas. El objetivo principal es explicar las formas de uso más interesantes de cada uno de los elementos, aportando el código que los acompaña. En el programa desarrollado en este proyecto, la interfaz se compone de:

43 “Edit Text”

36 “Static Text”

21 “Push Button”

3 “Pop-up menu”

2 “Radio button”

1 “Axis”

1.2.3.1. Definición de las propiedades por defecto

En la parte superior de cada ventana generada, se ofrece la posibilidad de definir las propiedades de inicio del programa. En la función llamada NombreVentana_OpeningFcn.

La definición de todas las estructuras se ha realizado en la ventana Inicio, también es en esta “*OpeningFcn*” donde se han asignado los valores iniciales a los flags que se utilizan en la lógica de las demás “*OpeningFcn*”.

```
function Inicio_OpeningFcn(hObject, eventdata, handles, varargin)

global  abrir entrar entrar1 entrar2 entrar3 cargar s index

global Datos Substrato Buffer Capa LimitesU LimitesL Optimizada
Representar DatosOptimizados
DatosOptimizados=0;
value0=0;

field1='L_Max';      field4='Rango';
field2='L_Min';      field5='S1_x';
field3='Paso';       field6='S1_y';
Datos=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0, field6,value0);

field4='d_capa';     field1='delta_capa';
field5='A_capa';     field2='gamma_capa';
field6='L_capa';     field3='lambda_gap_capa';
field7='P_capa';
Capa=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0 ,field6,value0, field7,value0);
LimitesU=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0 ,field6,value0, field7,value0);
LimitesL=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0 ,field6,value0, field7,value0);



field1='A_Sub';
field2='L1_Sub';
field3='P_s';
Substrato=struct(field1,value0, field2,value0, field3,value0);
```

```
field1='d_buff';    field3='L_buff';
field2='A_buff';    field4='P_buff';
Buffer=struct(field1,value0, field2,value0, field3,value0,
field4,value0);

field1='AO';    field5='A';
field2='delta';    field6='L';
field3='lambda_gap';    field7='P';
field4='d';
Optimizada=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0 ,field6,value0, field7,value0);
Representar=struct(field1,value0, field2,value0, field3,value0,
field4,value0, field5,value0 ,field6,value0, field7,value0);

s=0;
abrir=0;entrar=0;entrar1=0;entrar2=0;entrar3=0;cargar=0;index=0;

guidata(hObject, handles)

% Choose default command line output for Inicio
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
```

En los casos donde la ventana en la que se esté trabajando incluya bloques "*Edit Text*" se ha utilizado "*OpeningFcn*" para definir los valores que se representarán en estos bloques al abrir la ventana, dependiendo de los valores de los flags.

```
function Material_OpeningFcn(hObject, eventdata, handles, varargin)

global Capa LimitesU LimitesL entrar1 cargar abrir

if((entrar1==1)||(cargar==1))
    set(handles.edit24, 'String', Capa.gamma_capa);
set(handles.edit10, 'String', LimitesL.gamma_capa);
set(handles.edit19, 'String', LimitesU.gamma_capa);
    set(handles.edit25, 'String', Capa.delta_capa);
set(handles.edit11, 'String', LimitesL.delta_capa);
set(handles.edit15, 'String', LimitesU.delta_capa);
    set(handles.edit26, 'String',
Capa.lambda_gap_capa);set(handles.edit12, 'String',
LimitesL.lambda_gap_capa);set(handles.edit16, 'String',
LimitesU.lambda_gap_capa);
    set(handles.edit27, 'String', Capa.d_capa);
set(handles.edit13, 'String', LimitesL.d_capa);
set(handles.edit17, 'String', LimitesU.d_capa);
    set(handles.edit28, 'String', Capa.A_capa);
set(handles.edit14, 'String', LimitesL.A_capa);
set(handles.edit18, 'String', LimitesU.A_capa);
    set(handles.edit29, 'String', Capa.L_capa);
set(handles.edit20, 'String', LimitesL.L_capa);
set(handles.edit22, 'String', LimitesU.L_capa);
    set(handles.edit30, 'String', Capa.P_capa);
set(handles.edit21, 'String', LimitesL.P_capa);
set(handles.edit23, 'String', LimitesU.P_capa);
```

```
    entrar=1;
end
if((entrar1==0)&&(cargar==0)&&(abrir==0))
    set(handles.edit24, 'String', '');set(handles.edit10, 'String',
'');set(handles.edit19, 'String', '');
    set(handles.edit25, 'String', '');set(handles.edit11, 'String',
'');set(handles.edit15, 'String', '');
    set(handles.edit26, 'String', '');set(handles.edit12, 'String',
'');set(handles.edit16, 'String', '');
    set(handles.edit27, 'String', '');set(handles.edit13, 'String',
'');set(handles.edit17, 'String', '');
    set(handles.edit28, 'String', '');set(handles.edit14, 'String',
'');set(handles.edit18, 'String', '');
    set(handles.edit29, 'String', '');set(handles.edit20, 'String',
'');set(handles.edit22, 'String', '');
    set(handles.edit30, 'String', '');set(handles.edit21, 'String',
'');set(handles.edit23, 'String', '');
end
    % Update handles structure
guidata(hObject, handles);
```

En el caso de que sea la primera vez que se entra en esta ventana durante la ejecución del proceso de introducción de datos, se deben representar los bloques en blanco.

```
if((entrar1==0)&&(cargar==0)&&(abrir==0))
      set(handles.editX, 'String', '')
```

Si por el contrario ya se has cumplimentado los datos de los bloques "*Edit Text*" con anterioridad o se han cargado unos datos de una ejecución anterior del programa, se representan estos datos en los bloques, poniendo 0 en aquellos parámetros que no se hubiesen introducido con anterioridad.

```
if((entrar1==1)||(cargar==1))
      set(handles.editX, 'String', Capa.gamma_capa);
```

1.2.3.2. Edit Text

Este bloque ha sido utilizado en su totalidad para el almacenamiento de cadenas de caracteres introducidas por el usuario.

```
function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Datos entrar
entrar=1;
Datos.L_Max=str2double(get(hObject, 'String'));
if isnan(Datos.L_Max)
    errordlg('El valor debe ser numérico', 'ERROR')
    set(hObject, 'String', '');
```

```
    Datos.L_Max=0;
end

if (Datos.L_Max<=Datos.L_Min)
    errordlg('El valor del margen superior debe ser mayor al valor del
margen inferior', 'ERROR')
    set(hObject, 'String', '');
    Datos.L_Max=0;
end
guidata(hObject, handles);
```

En el caso de que la información que demandamos al usuario quiera ser interpretada como cadena de caracteres utilizamos

```
   Texto=(get(hObject, 'String'));
```

Almacenando la cadena de caracteres que introduce el usuario como tipo '*String*', en la variable Texto

Si por otro lado, esa información se quiere procesar como un valor numérico utilizamos

```
Datos.L_Max=str2double(get(hObject, 'String'));
```

Con esta sentencia se almacena la cadena que introduce el usuario como un valor numérico, mediante la transformación str2double(), en el campo "L_Max" de la estructura "Datos".

En este caso se utilizan los campos "L_Max" y "L_min" de la estructura "Datos" para acotar el margen de longitudes de onda con el que se desea trabajar, por lo que sí el límite superior "L_Max" es menor al límite inferior "L_min" el programa nos dará el mensaje de error "`El valor del margen superior debe ser mayor al valor del margen inferior`" y representará un 0 en el bloque mediante la sentencia

```
if (Datos.L_Max<=Datos.L_Min)
    errordlg('El valor del margen superior debe ser mayor al valor del
margen inferior', 'ERROR')
    set(hObject, 'String', '');
    Datos.L_Max=0;
end
```

En el caso de que la cadena de caracteres introducida por el usuario no fuese una cadena numérica se mostraría el mensaje de error "`El valor debe ser numérico`" y dejará el bloque en blanco, asignando un 0 a la variable "L_Max" mediante la sentencia

En este caso, es en el *callback* del propio elemento donde se almacena el valor de la variable de manera instantánea. Cuando el usuario termina de introducir la cadena de caracteres, esta quedará almacenada en la variable asociada.

Esta forma de utilización de "*Edit Text*" es interesante para por ejemplo representar en los ejes la variación de las funciones que produce la modificación de un parámetro en concreto.

Por otro lado, en otro tipo de aplicación puede interesar que todas las variables queden almacenadas de golpe al realizar la activación de otro elemento, como por ejemplo un "*Push Button*", de manera que hasta que el usuario no valide la acción pulsando dicho elemento, no se sobrescriba ninguna variable.

1.2.3.3. Push Button

Este bloque permite la ejecución de una subrutina tras hacer "*click*" en el botón y se ha utilizado de diversas formas. Entre ellas las más interesantes son las nombradas en los tres siguientes puntos

1.2.3.3.1. Introducción de datos tras activación

Supóngase que una función ha sido pensada para realizar una operación cuando se hayan introducido todos los datos en los "*Edit Text*", por ejemplo la optimización.

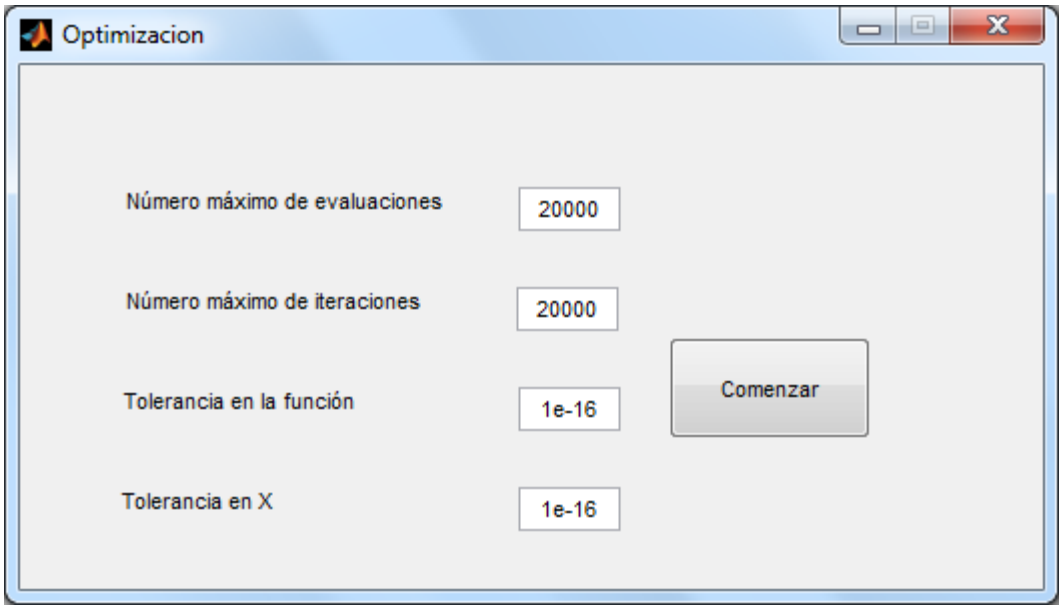


Figura 2.4 Ventana de parametrización de la optimización

En este caso lo lógico será que el almacenamiento de los datos se realice cuando el usuario esté listo para comenzar la optimización, por lo que el código se ha estructurado de la siguiente forma.

```
function edit1_Callback(hObject, eventdata, handles)

if isnan(hObject)
    errordlg('El valor debe ser numérico', 'ERROR')
    set(hObject, 'String', '');
    hObject=0;
end

global index
index=1;

function edit2_Callback(hObject, eventdata, handles)

if isnan(hObject)
    errordlg('El valor debe ser numérico', 'ERROR')
    set(hObject, 'String', '');
    hObject =0;
end

global index
index=1;

function edit3_Callback(hObject, eventdata, handles)

if isnan(hObject)
    errordlg('El valor debe ser numérico', 'ERROR')
    set(hObject, 'String', '');
    hObject =0;
end

global index
index=1;

function pushbutton1_Callback(hObject, eventdata, handles)

global x0 lb ub s Datos Capa LimitesL LimitesU Optimizada
global FunMax IterMax FunTol XTol
.
.
.
FunMax=str2double(get(handles.edit1,'String'));
IterMax=str2double(get(handles.edit2,'String'));
FunTol=str2double(get(handles.edit3,'String'));
XTol=str2double(get(handles.edit4,'String'));
.
.
.
```

En esta ventana, en cada uno de los "*Edit Text*" se le da valor 1 al flag de nombre "index", este flag indica que los bloques han sido modificados por el usuario, de manera que en posteriores aperturas de la ventana Optimización, se represente el valor asociado a su "*handler*",

```
function editX_Callback(hObject, eventdata, handles)
```

```
if isnan(hObject)
    errordlg('El valor debe ser numérico', 'ERROR')
    set(hObject, 'String', '');
    hObject =0;
end

global index
index=1;
```

El almacenamiento de todas las variables introducidas por el usuario se hace cuando pulsa el botón "Comenzar", utilizando los "*handlers*" asociados a cada elemento

```
function pushbutton1_Callback(hObject, eventdata, handles)
.
.
.
FunMax=str2double(get(handles.edit1,'String'));
IterMax=str2double(get(handles.edit2,'String'));
FunTol=str2double(get(handles.edit3,'String'));
XTol=str2double(get(handles.edit4,'String'));
.
.
.
```

1.2.3.3.2. Guardado de estructuras

Es interesante tener la posibilidad de almacenar los datos introducidos y obtenidos tras la ejecución del programa, para poder acceder a ellos tanto en otros formatos como pueden ser (*.dat), (*.mat) o (*.txt), para tener la posibilidad de ejecutar posteriormente esa información desde el programa en posteriores usos.

Con una sencilla función se pueden guardar las estructuras que han ido almacenándose y creándose en la ejecución del programa

```
function pushbutton6_Callback(hObject, eventdata, handles)

global Capa LimitesU LimitesL Substrato Buffer Datos s Optimizada
n_capa n_buf abs_capa
save(uiputfile({'*.mat'},'File Selector'),'n_capa', 'n_buf',
'abs_capa','Capa','s','Optimizada','LimitesU','LimitesL','Datos',
'Substrato', 'Buffer')
```

La función de MATLAB "*save*" unida con la función "*uiputfile*" creará un archivo del tipo especificado, en este caso (*.mat), que almacenará las variables y estructuras pasadas por parámetro. Abrirá una ventana de búsqueda "*File Selector*", donde se puede sobrescribir o crear un nuevo fichero con la extensión (*.mat)

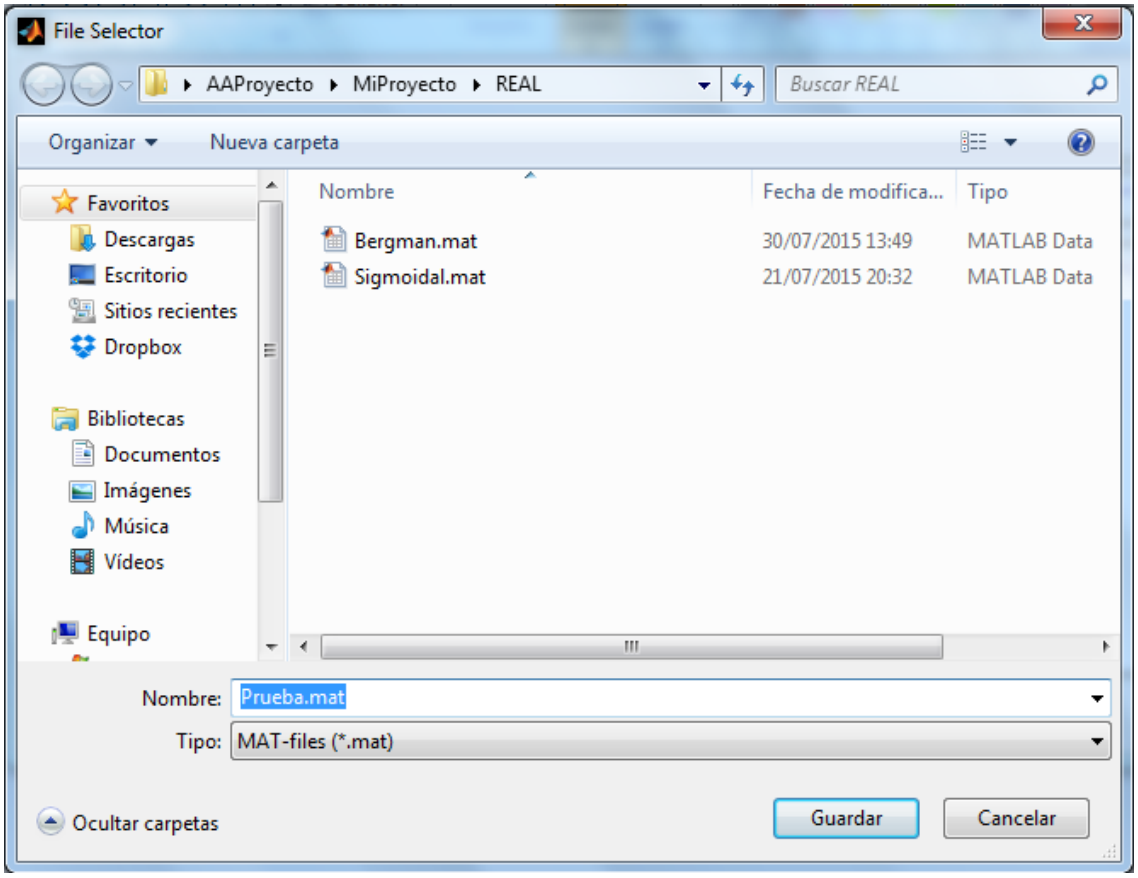


Figura 2.5 Guardado de estructuras

Otra de las opciones implementadas ha sido, la creación de un documento (*.txt) en el que el usuario pueda definir la cabecera del mismo. Tras una correcta optimización y ajuste el usuario puede querer almacenar la información representada en las gráficas de

- Índice de refracción.
- Absorción.
- Transmisión calculada.
- Transmisión calculada y experimental.
- Para comprobar la información que ahí se está representando

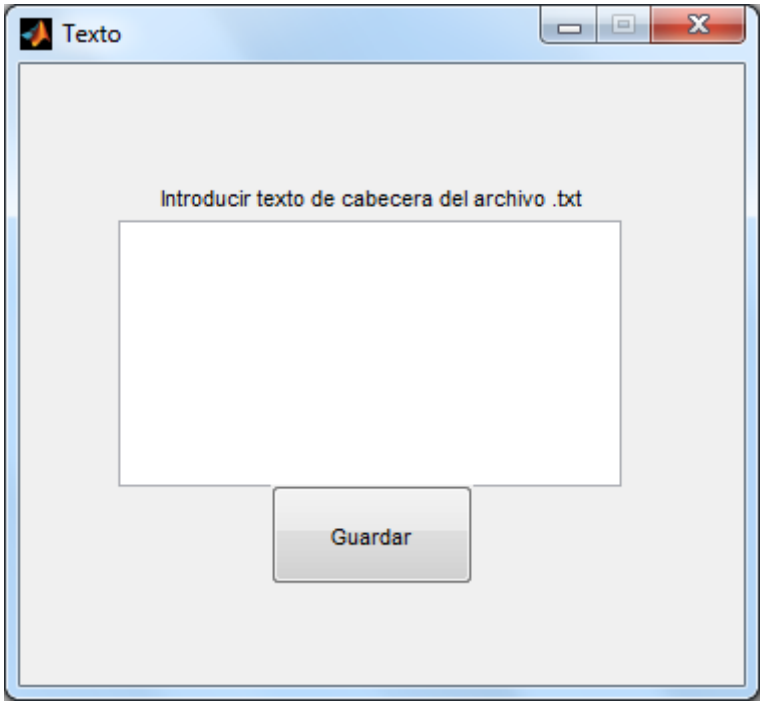


Figura 2.6 Guardado del archivo .txt

Para ello se ha implementado el siguiente código

```
function edit1_Callback(hObject, eventdata, handles)

global texto
texto=get(hObject,'String');
.
.
.
function pushbutton1_Callback(hObject, eventdata, handles)

global texto Datos n_capa abs_capa Trans_1 TT
i=1;j=1;
a=size(Datos.Rango);
for i=a:-1:1
    A(j)=Datos.Rango(i);
    j=j+1;
    A(j)=n_capa(i);
    j=j+1;
    A(j)=abs_capa(i);
    j=j+1;
    A(j)=Trans_1(i);
    j=j+1;
    A(j)=TT(i);
    j=j+1;
    i=i+1;
end
[FileName,PathName] = uiputfile({'*.txt'},'File Selector');
save(FileName);
fileID = fopen(FileName,'w');
[nrows,ncols]=size(texto);
for row=1:nrows
    fprintf(fileID,'%s\n',texto{row,:});
end
fprintf(fileID,'\r\n');
```

```
fprintf(fileID, '%6s %12s %12s %12s
%12s\r\n','Rango','n_capa','abs_capa','Trans_1','TT');
fprintf(fileID,'%6.2f %12.8f %12.8f %12.8f %12.8f\r\n',A);
fclose(fileID);
close Texto
```

En la función del "*Edit Text*" se almacena la cadena introducida por el usuario en formato '*String*' en la variable Texto. Esta cadena será la que aparecerá como cabecera del documento (*.txt).

```
i=1;j=1;
a=size(Datos.Rango);
for i=a:-1:1
    A(j)=Datos.Rango(i);
    j=j+1;
    A(j)=n_capa(i);
    j=j+1;
    A(j)=abs_capa(i);
    j=j+1;
    A(j)=Trans_1(i);
    j=j+1;
    A(j)=TT(i);
    j=j+1;
    i=i+1;
end
```

Todos los *arrays* que se van a representar tienen la misma dimensión, que será el número de elementos que forman el margen de longitudes de onda introducido en la ventana Datos, por lo que el tamaño de ese *array* se almacena en la variable 'a'. Para representar los datos por columnas un bucle "*for*" recorrerá los 4 *arrays* por ordinales, de manera que queden almacenados en el *array* A los elementos de la siguiente forma

A=[Datos.Rango(1), n_capa(1), abs_capa(1), Trans_1(1), TT(1), Datos.Rango(2), n_capa(2), abs_capa(2), Trans_1(2), TT(2)… Datos.Rango(a), n_capa(a), abs_capa(a), Trans_1(a), TT(a)]

```
[FileName,PathName] = uiputfile({'*.txt'},'File Selector');
save(FileName);
fileID = fopen(FileName,'w');
```

En este fragmento de código se crea el fichero de texto con el nombre que el usuario introduzca en el "File Selector", posteriormente se guarda el identificador de ese fichero recién creado en la variable '*fileID*' y se abre con permisos de escritura.

```
 [nrows,ncols]=size(texto);
for row=1:nrows
    fprintf(fileID,'%s\n',texto{row,:});
end
fprintf(fileID,'\r\n');
```

```
fprintf(fileID, '%6s %12s %12s %12s
%12s\r\n','Rango','n_capa','abs_capa','Trans_1','TT');
fprintf(fileID,'%6.2f %12.8f %12.8f %12.8f %12.8f\r\n',A);
fclose(fileID);
close Texto
```

Lo primero que hace el fragmento de código superior es determinar el tamaño de la cadena introducida en "*Edit Text*" y almacenarlo en la variable '*nrows'*, posteriormente recorriendo el bucle '*for'* se escribe en el fichero letra a letra dicha cadena hasta su final.

```
fprintf(fileID, '%6s %12s %12s %12s
%12s\r\n','Rango','n_capa','abs_capa','Trans_1','TT');
```

posteriormente el programa va accediendo a cada uno de los elementos que integran el *array* 'A', y escribe estos elementos en el fichero, de manera que quedan ordenados por columnas , por eso el formato del *array* 'A', `'Rango','n_capa','abs_capa','Trans_1','TT'` .

```
fprintf(fileID,'%6.2f %12.8f %12.8f %12.8f %12.8f\r\n',A);
fclose(fileID);
close Texto
```

Finalizamos cerrando el documento y la ventana "Texto". Consiguiendo un documento con la siguiente estructura:

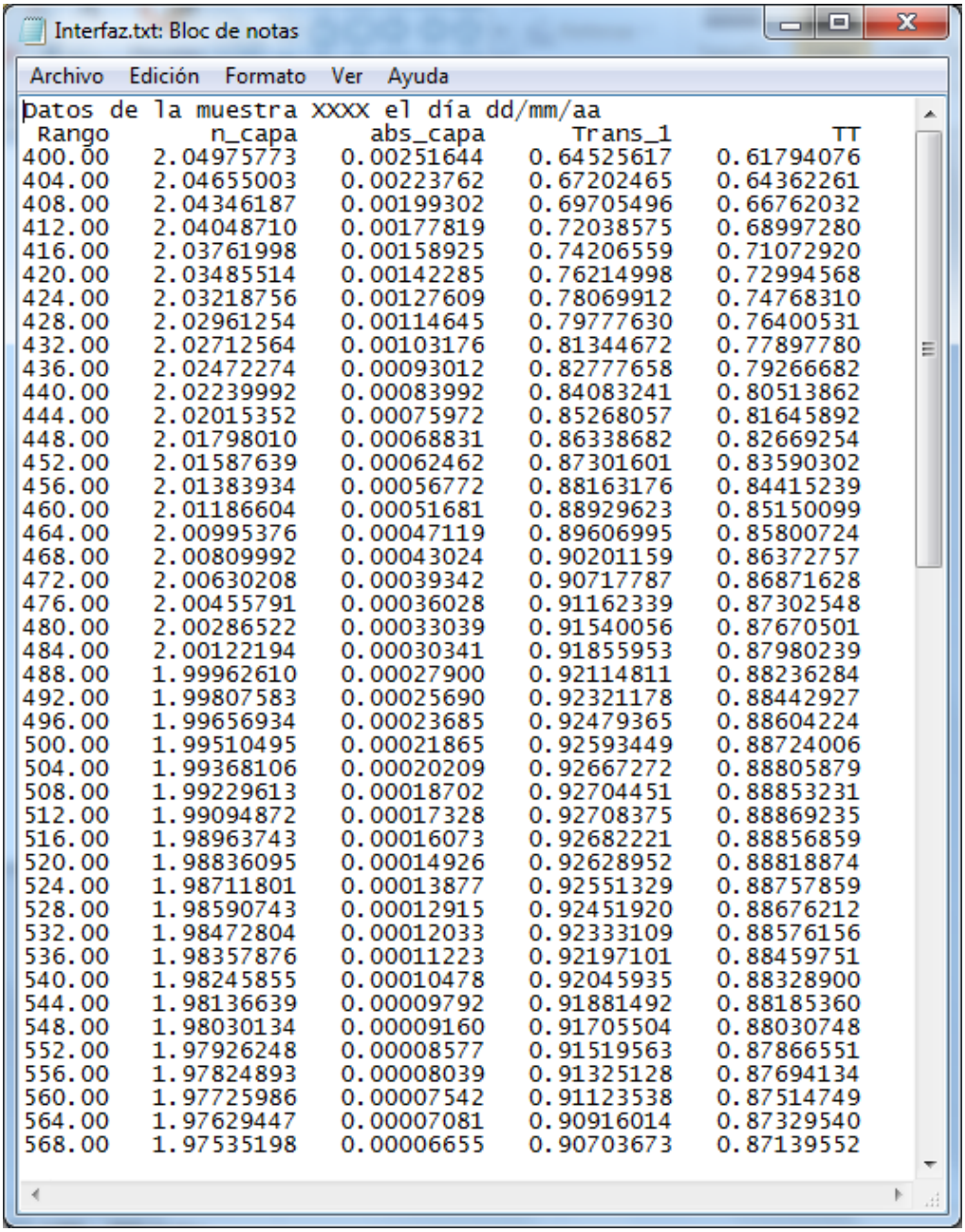


Figura 2.7 Almacenamiento de datos en formato .txt

1.2.3.3.3. Carga de estructuras

Para poder acceder a los datos, estructuras y resultados obtenidos con anterioridad, se ha desarrollado una funcionalidad de carga de datos, lo que se realiza muy fácilmente utilizando la siguiente función.

```
function pushbutton10_Callback(hObject, eventdata, handles)

global cargar
cargar=1;

load(uigetfile({'*.mat'},'File Selector'))
```

La función *uigetfile* sobrescribe en el programa todas las estructuras, poniendo en ellas los valores de los campos que permanecían guardadas en el fichero (*.mat).

El flag "cargar" se utiliza para representar, en los bloques "*Edit Text*", estas antiguas variables.

1.2.3.4. Pop-up Menu

El bloque "*Pop-up Menu*" basa su funcionamiento en una sentencia "*switch-case*" con tantos casos como campos se definan en el "*property inspector*" de este elemento.

En esta interfaz se ha utilizado para elegir las funciones con las que se quieren realizar el ajuste de los datos y para seleccionar las gráficas que se quieren representar en el elemento "*Axis*".

```
function popupmenu2_Callback(hObject, eventdata, handles)

global Datos Optimizada Sigmoidal Bergman usuario
global abs_capa

 a=get(hObject,'Value');

switch a

    case 1

        E=1239./Datos.Rango;
        Eg=1239/Optimizada.lambda_gap;
        deltaE=Optimizada.delta;
        abs_capa=Optimizada.AO./(1+(exp((Eg-E)./deltaE)));
        Sigmoidal=1;Bergman=0;usuario=0;

    case 2

abs_capa=Optimizada.AO+Optimizada.delta*exp(Optimizada.lambda_gap./Dat
os.Rango);
        disp(abs_capa)
        Sigmoidal=0;Bergman=1;usuario=0;

    case 3

        Sigmoidal=0;Bergman=0;usuario=1;
        IndiceAbsorcion
        uiwait

end
```

Al seleccionar uno de los campos la variable “a” almacenará el identificador de dicho campo, de manera que se active el caso demandado por el usuario. En este ejemplo:

1 Sigmoidal

2 Bergman (Urbach tail)

3 Usuario

En los casos 1 y 2 se ejecutarán las funciones correspondientes a la ecuación que representan. Almacenando en variables globales los datos de la absorción de la capa, en función de los parámetros característicos del material llevado a estudio.

En el caso 3 se abrirá la ventana donde podremos introducir por código, de la misma manera que en Bergman y en Sigmoidal, la función que consideremos apropiada para realizar la caracterización del material.

Las variables Sigmoidal, Bergman y Usuario tienen su flag correspondiente, que indica al programa que función ha sido seleccionada para en posteriores bloques utilizar las funciones correspondientes.

```
global Datos Buffer abs_capa s n_capa n_buf Representar TT Trans_1
a=get(hObject,'Value');
set(handles.radiobutton1, 'Value', 0);
set(handles.radiobutton2, 'Value', 0);
Lc=Datos.Rango.^2;

C1=(n_capa+s).*(1+n_capa);
C2=(n_capa-s).*(1-n_capa);
C3=((n_buf+(n_capa.*s./n_buf)).*(1+n_capa));
C4=(((n_capa.*s./n_buf)-n_buf).*(1-n_capa));
Desf_capa=(2*pi*Representar.d)*(n_capa./Datos.Rango);
Desf_buf=(2*pi*Buffer.d_buff)*(n_buf./Datos.Rango);

x=exp(-abs_capa*Representar.d);
CO=cos(2*Desf_capa);
Num=16*s.*(x.*(n_capa.^2)); % n1^2 segun paper
A=(C1.^2+((C2.*x).^2)+2*x.*C1.*C2.*CO).*((cos(Desf_buf)).^2);
B=(C3.^2+((C4.*x).^2)+2*x.*C3.*C4.*CO).*((sin(Desf_buf)).^2);
B2=(C1.*C4-C2.*C3).*x.*sin(2*Desf_capa).*sin(2*Desf_buf);

Trans_1=Num./(A+B+B2);
T_s=(4*s)./((1+s).^2);
T_Aux=Trans_1.*T_s;
TT=T_Aux./((1-x)+Trans_1+x.*T_s-T_Aux);

switch a
    case 1
```

```
        plot (Datos.Rango,n_capa,'k');
        hold on
        plot (Datos.Rango,s,'b');

        legend('calculado','experimental')
        xlabel('nm(lhanda)')
        ylabel('refracción')
        hold off

    case 2

        plot (Datos.Rango,abs_capa,'b');
        xlabel('nm(lhanda)')
        ylabel('nm-1(alpha)')
        hold off

    case 3

        plot (Datos.Rango,Trans_1,'b');

        xlabel('nm(lhanda)')

    case 4

        plot (Datos.S1_x,Datos.S1_y,'b');ylim([0 1]);
        hold on
        plot (Datos.Rango,TT,'k');
        legend('experimental','calculado')
        xlabel('nm(lhanda)')
        hold off

end
guidata(hObject, handles);
```

1.2.3.5. Radio Selector

El "*Radio Selector*" es un bloque que proporciona el valor de la condición de una sentencia "*if*", en casos puntuales es muy ventajoso utilizar estos bloques, sobre todo a la hora de definir condiciones que afectarán a otros bloques concatenados.

En esta interfaz solamente se han utilizado 2 bloques "*Radio Selector*", pero han facilitado notablemente la lógica de representación de funciones en el elemento "*Axis*". Ambos operan de la misma forma.

```
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Representar Optimizada
valor=get(hObject,'Value');
if valor==1
    set(handles.radiobutton2, 'Value', 0);
    if (Optimizada.lambda_gap==0)
    errordlg('Los datos no estaban optimizados', 'ERROR')
```

```
    set(hObject, 'Value', 0);
    end
    if (Optimizada.lambda_gap>0)
    Representar.AO=Optimizada.AO;
    Representar.delta=Optimizada.delta;
    Representar.lambda_gap=Optimizada.lambda_gap;
    Representar.d=Optimizada.d;
    Representar.A=Optimizada.A;
    Representar.L=Optimizada.L;
    Representar.P=Optimizada.P;
    end
end
```

Si el selector está activo (verde), entonces su "*handler*" obtiene valor 1, este valor queda asignado a la variable "valor" y en caso de ser 1 realiza la sentencia "*if*", en la que primero se deshabilita el otro selector presente en la interfaz, asignándole un 0 a su "*handler*", seguidamente se comprueba si la estructura, que alberga los datos optimizados que se quieren representar, está llena. En caso de no estarlo se muestra un mensaje de error y se asigna un 0 al "*handler*" deshabilitando este selector. Por el contrario sí si hay datos en esta estructura, se pasarían uno a uno a la estructura "Representar", esta es la estructura con la que se trabaja en el elemento "*Axis*", de manera que se borran los datos que ahí podría haber de una representación anterior.

1.2.3.6. Función lsqnonlin

Para ajustar las propiedades ópticas de la capa analizada, se han de ajustar los resultados teóricos a los datos de la transmisión obtenidos con el espectrofotómetro, mediante mínimos cuadrados, para ello se ha empleado la función "lsqnonlin"

```
options = optimset('lsqnonlin');
optnew =
optimset(options,'MaxFunEvals',FunMax,'MaxIter',IterMax,'TolFun',FunTo
l,'TolX',XTol);

[p,resnorm,residual,exitflag] =
lsqnonlin(@funOptimizacion,x0,lb,ub,optnew);% Compute function values
Optimizada.AO=p(1);
Optimizada.delta=p(2);
Optimizada.lambda_gap=p(3);
Optimizada.d=p(4);
Optimizada.A=p(5);
Optimizada.L=p(6);
Optimizada.P=p(7);

close Optimizacion
```

En la variable de nombre *optnew* se almacenan las condiciones de contorno con las que realizamos el ajuste

- MaxFunEvals: Máximo número de funciones a evaluar.
- MaxIter: Máximo número de iteraciones que se quieren realizar.
- TolFun: Tolerancia en la función.
- TolX: Tolerancia en los valores de la variable x0.

p = lsqnonlin(fun,x0,lb,ub,options) En lb y ub se establecen los limites en los que se quiere ajustar la variable x0, lb ≤ x ≤ ub.

Entonces se realizará un ajuste por mínimos cuadrados utilizando como modelo, la estructura generada en la función a la que llama, en nuestro caso "funOptimizacion". Una vez realizado el ajuste almacenará los resultados obtenidos en el *array* "p" del cual obtendremos nuestros resultados de la caracterización, estos se almacenarán en la estructura de nombre Optimizada, para su posterior uso en representaciones.

1.2.3.7. Función de optimización

Para poder realizar el ajuste por mínimos cuadrados y obtener los parámetros ópticos del material llevado a estudio, es necesario comparar los datos de la transmisión con una referencia, dicha referencia se genera en el fichero "funOptimizacion" al que llama la función "lsqnonlin". En este fichero se obtiene la transmisión experimental de la capa.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ecuaciones de Bergman et al.                    %
% por F. B. Naranjo Junio 2006                    %
% modificación por Alejandro Villalba             %
% para TFG interfaz gráfica Junio 2015            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function F = fun(x)

global s Sigmoidal Bergman usuario
global Datos abs_capa
global i
format long;

i=i+1;

L=Datos.Rango;
%size (L)
```

```
%parámetros de entrada

A0=x(1);
delta=x(2);
lambda_gap=x(3);
d_capa=x(4);
A_capa=x(5);
L_capa=x(6);
P_capa=x(7);

Lc=L.^2;
n_capa=(P_capa+((A_capa*Lc)./(Lc-(L_capa)^2))).^(1/2);
```

Primero se asignan los elementos del *array* de la capa, pasados desde la función *lsqnonlin*, a las variables que se utilizan en esta sección del programa.

```
%%%sigmoidal
if (Sigmoidal==1)
    E=1239./L;
    Eg=1239/lambda_gap;
    deltaE=delta;
    abs_capa=A0./(1+(exp((Eg-E)./deltaE)));

end

if(Bergman==1)
    abs_capa=A0+delta*exp(lambda_gap./Datos.Rango);
end

if(usuario==1)
    abs_capa;
end

Calculamos la absorción de la capa dependiendo de las
ecuaciones que el usuario haya decidido utilizar
%% Transmisión a través de aire/capa

C1=(n_capa+s).*(1+n_capa);
C2=(n_capa-s).*(1-n_capa);

Desf_capa=(2*pi*d_capa)*(n_capa./L);
x=exp(-abs_capa*d_capa);
CO=cos(2*Desf_capa);
Num=16*s.*x.*(n_capa.^2); % n1^2 segun paper

A=(C1.^2+((C2.*x).^2)+2*x.*C1.*C2.*CO);
Trans_1=Num./A;

% Transmisión del substrato

T_s=(4*s)./((1+s).^2);

% Transmisión total
```

```
T_Aux=Trans_1.*T_s;
TT=T_Aux./((1-x)+Trans_1+x.*T_s-T_Aux);

Obtenemos la transmisión a través del aire y la capa y la
transmisión del substrato, con ambas determinamos la
transmisión total.

Tam=size(Datos.S1_x);
iteracion=1;
for iteracion=1:Tam(1)
    if(Datos.S1_x(iteracion)==Datos.L_Min)
        MargenA=iteracion;
        iteracion=1;
    end
end
for iteracion=1:Tam(1)
    if(Datos.S1_x(iteracion)==Datos.L_Max)
        MargenB=iteracion;
        iteracion=1;
    end
end

F=abs((Datos.S1_y(MargenB:MargenA)-TT));
```

Se utiliza el margen de longitudes de onda que el usuario haya determinado desde la ventana Datos, para realizar los ajustes de la transmisión total, en esa región. Los datos obtenidos, se almacenan en el *array* F y se envía a la función *lsqnonline*, para que realice el ajuste por mínimos cuadrados con esta información.

1.3. RESULTADOS Y CONCLUSIONES

En este apartado se detallan los resultados prácticos obtenidos mediante la herramienta desarrollada y las conclusiones que se extraen de su funcionamiento.

Los siguientes resultados se obtuvieron para láminas de óxido multicomponente con diferentes concentraciones químicas, fabricadas mediante la técnica de *co-sputtering*.

En el caso del compuesto U049, que tras realizar las medidas, presentaba las siguientes concentraciones:

- (0.46 ± 0.07) %wt Al_2O_3:
- (9.3 ± 1.4) %wt ZnO
- (85 ± 2) %wt In_2O_3:
- (5.6 ± 0.4) %wt SnO_2

Se obtuvieron los resultados mostrados en la (figura 3.1)

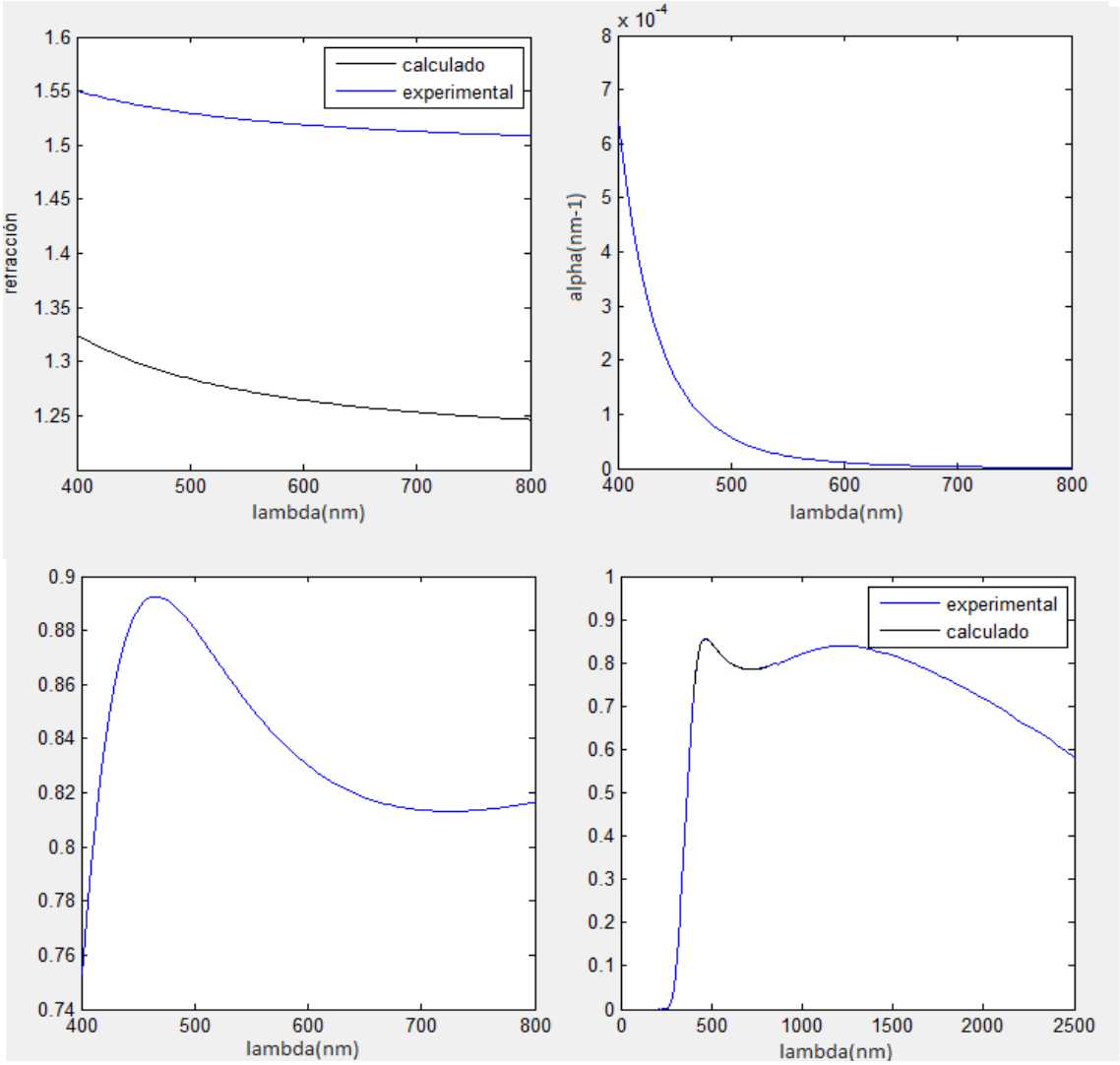


Figura 3.1 Gráficas obtenidas para la muestra TLUV049

Como se puede ver en la cuarta gráfica de la **(figura 3.1)** el ajuste realizado con la herramienta es muy bueno, resultando una superposición de las gráficas de la transmisión obtenida mediante el ensayo con el espectrofotómetro y la calculada con la herramienta, para el margen de longitudes de onda (400, 800)nm.

Los datos de la caracterización de la capa se muestran en la (**figura 3.2)**

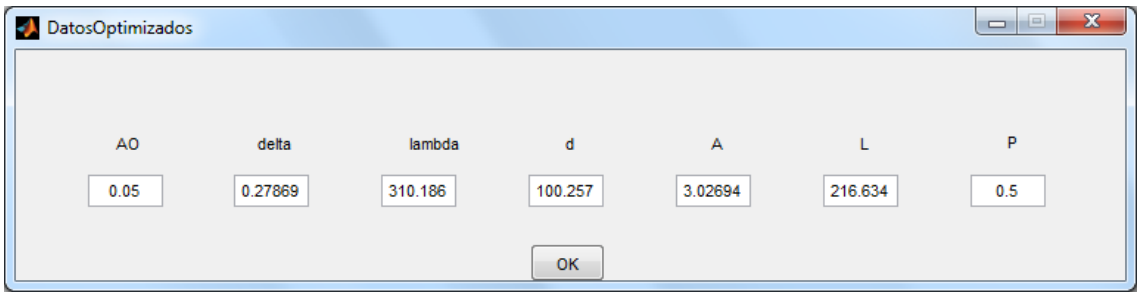


Figura 3.2 Parámetros de la capa obtenidos para la muestra TLUV049

La otra muestra llevada a estudio ha sido el U059, dicha muestra presenta la siguiente composición:

(0.24 ± 0.04) %wt Al_2O_3:

(7.9 ± 0.7) %wt ZnO

(85 ± 1) %wt In_2O_3:

(6.6 ± 0.5) %wt SnO_2

Con ella se obtuvieron los resultados mostrados en la **(figura 3.3)**

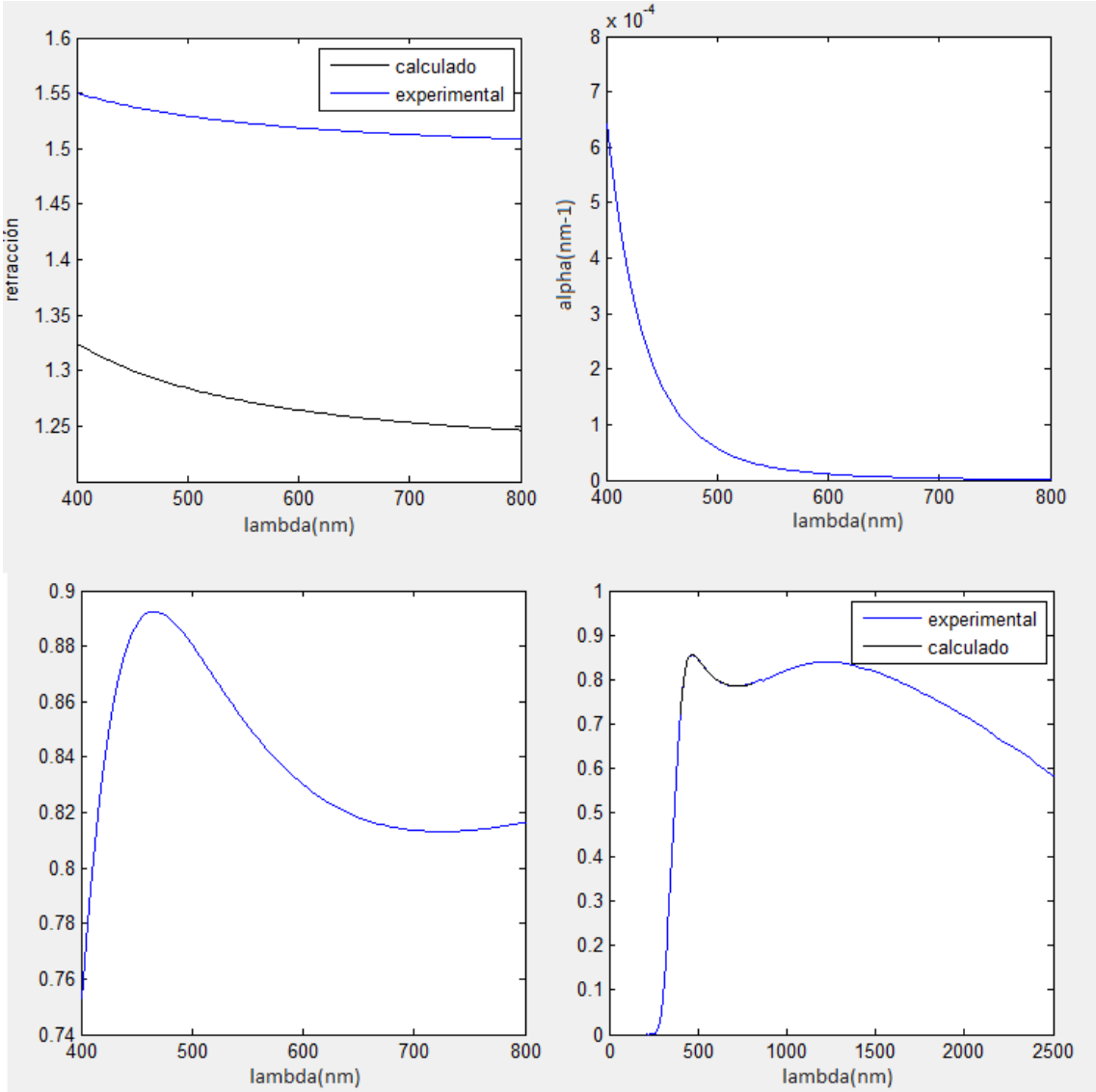


Figura 3.3 Gráficas obtenidas para la muestra TLUV059

En este caso se eligió un margen de longitudes de onda más amplio (320, 800)nm e igualmente, como podemos apreciar, el ajuste también resulta muy bueno.

Los datos de la caracterización de la capa se muestran en la (**figura 3.4)**

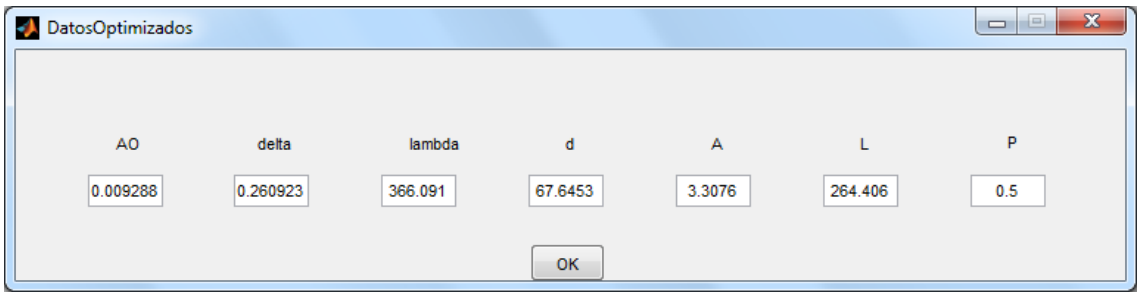


Figura 3.4 Parámetros de la capa obtenidos para la muestra TLUV059

Para concluir, cabe destacar la potencia que MATLAB proporciona, gracias a la cual, se ha podido desarrollar esta interfaz que realiza un ajuste, compuesto por en torno a mil iteraciones formadas por diversas ecuaciones y complejos cálculos, en un tiempo mínimo de 0'308 segundos y un tiempo máximo de 0'423 segundos, para las 20 muestras realizadas en el margen de longitudes de onda (400, 800)nm, dejando un residuo que en ningún caso superó 0.001% respecto de la transmisión experimental.

Un inconveniente es que al ampliar mucho el margen de longitudes de onda y al trabajar en longitudes de onda altas, el ajuste que realiza el programa es de peor calidad, ya que en este caso intervienen otros parámetros que no se han tenido en cuenta en las ecuaciones que vienen por defecto.

Para solventar este pequeño inconveniente, la interfaz permite el ajuste por tramos, de manera que si el usuario quisiera estudiar los efectos relacionados con una elevada concentración de portadores, observables a través de una resonancia de plasmón a longitudes de onda largas, tiene la opción de seleccionar el margen de longitudes de onda de interés , pudiendo, a su vez, implementar las ecuaciones de absorción e índice de refracción, que le permitan llevar a cabo este estudio, en la ventana de usuario

1.3.1. Trabajo futuro

En este punto se detallan aquellos aspectos mejorables de la herramienta, que el desarrollador y el tutor han identificado. Aunque la herramienta cumple muy bien con sus objetivos, hay ciertos puntos en los que resultaría muy interesante introducir ciertas mejoras.

Para lograr una herramienta con mayores posibilidades, resultaría muy beneficioso ampliar los modelos y ecuaciones que se le ofrecen al usuario. A parte, ofrecer la posibilidad de que aquellos modelos introducidos por el usuario, quedasen almacenados de una manera sencilla, para futuras utilizaciones.

Otro punto a tener en cuenta, es la reducción del código generado. Para reducir el código significativamente podría crearse un fichero que incluyese algunas funciones de adquisición de datos. Los bloques "*Edit Text*" podrían llamar a estas funciones dentro de la subrutina y que estas asignasen el valor a la variable del bloque, sin tener que programar la adquisición una y otra vez en los casi 50 bloques "*Edit Text*"

Esta herramienta trabaja con el espectro de transmisión del semiconductor. La introducción del análisis de espectros de reflexión y su combinación con los de transmisión mejoraría el margen de error en los coeficientes de absorción e índice de refracción obtenidos.

Una significativa mejora, sería introducir una nueva lógica en la opción de optimización, que permitiese elegir qué tipo de datos quieren optimizarse (datos optimizados, o datos sin optimizar). De este modo el usuario que haya obtenido un ajuste poco satisfactorio, no tendría que introducir los datos optimizados en los datos de la capa para volver a optimizarlos.

2. DIAGRAMAS


Lógica de la interfaz
Si
Si
Carga datos
¿Están optimizados?
Inicio
¿Usar datos anteriores?
No
No
Si
Guarda los datos
Introduce datos
Elige modelos
Optimiza
Elige modelos
Representa
¿Son válidos los datos?
No

3. PRESUPUESTO

Este capítulo proporciona información detallada acerca de los costes teóricos del desarrollo del proyecto, incluyendo los costes de materiales y las tasas profesionales.

3.1. COSTES MATERIALES

Objeto		*Cantidad*	*Precio unidad (€)*	*Precio total (€)*
Hardware	Portátil	1	600 €	600 €
Software	Microsoft Windows 8.1	1	0 €	0 €
	MATLAB-Student License	1	125 €	125 €
	Microsoft Office Home and student	1	100 €	100 €
TOTAL				825 €

3.2. TASAS PROFESIONALES

Objeto	*Periodo en meses*	*Salario mensual (€)*	*Coste total (€)*
Ingeniería	4	1.300 €	5.200 €
Total			5.200 €

3.3. COSTES TOTALES

Los costes totales del proyecto han sido obtenidos sumando los costes materiales y profesionales aplicándose el IVA. Además hay que tener en cuenta los costes de impresión y encuadernación.

Objeto		***Costes totales***
Costes materiales		825 €
Costes profesionales		5.200 €
Proyecto	Impresión	40 €
	Encuadernación	100 €
Subtotal		6.165 €
IVA (21%)		1.295 €
TOTAL		7.460 €

Los costes totales del proyecto ascienden a siete mil cuatrocientos sesenta euros.

4. MANUAL DE USUARIO

En esta sección se muestra la descripción de la interfaz desarrollada. Todo usuario requiere un tiempo de adaptación a una interfaz, cuanto mejor haya sido desarrollada la interfaz menor será el tiempo dedicado para realizar esta acción. Una buena interfaz se vuelve invisible a la hora de la interacción con el usuario, éste se dará cuenta de su existencia cuando observe algún error o mal diseño quedándose perdido y sin saber cuál es el siguiente paso a realizar. Por el contrario el usuario debe sentirse seguro y confiado haciendo que él mismo tenga el control total de la herramienta.

Desde un principio se tuvo la idea de crear una interfaz sencilla, clara, predecible y fácil de usar a la vez que práctica para el aprendizaje de manera interactiva y útil.

Todo diseño debe empezar por la comprensión de quienes son los usuarios de destino. Desde el primer boceto del programa se tuvo claro su filosofía, se decidió realizar el diseño en varias ventanas de manera que la interfaz no quedase muy cargada y ayudase al usuario primerizo a la total comprensión del procedimiento a seguir. Además se implementaron ayudas como mensajes de error para facilitar la solución inmediata

4.1. OBJETIVOS DEL PROGRAMA

El propósito general de la interfaz es La caracterización óptica de semiconductores mediante los datos del espectro de transmisión de la capa. A partir de este espectro y suponiendo unas relaciones para los parámetros ópticos preestablecidas para este tipo de materiales, se pueden obtener las propiedades ópticas de las capas en función de la longitud de onda. Ahora bien, la obtención de los parámetros ópticos que determinan las propiedades de las capas exige un proceso de ajuste por mínimos cuadrados de los datos experimentales a una función prueba Es por tanto de inmediata aplicación el desarrollo de una herramienta para la realización de estos ajustes con una apropiada interfaz de usuario, de forma que se pueda utilizar en entornos de investigación y desarrollo sin necesidad de conocimientos de algoritmos de ajuste y programación.

En este capítulo se detallará un manual para el usuario, con el cual se podrá acceder a cualquier funcionalidad de la aplicación. Se explicará detalladamente de qué forma introducir los datos para que el programa realice los cálculos correctamente y de qué forma, tras obtener un resultado satisfactorio extraer las estructuras de datos y la información relevante de dichos resultados.

4.2. MANUAL DE USUARIO

En esta sección se pretende presentará toda la información relevante y necesaria para el uso, el control y la comprensión de la interfaz gráfica. A medida que se avance en los siguientes apartados el usuario podrá comprender de lo que es capaz la interfaz así como los componentes de la interfaz que necesita utilizar para obtener la representación que desea. Además, se ha realizado un ejemplo en el que se podrá ver como introducir los datos y los resultados que se obtienen tras una correcta ejecución del programa.

Previamente a este apartado será interesante consultar el diagrama de flujo del sistema que queda adjunto al final de la memoria. De tal manera que antes de entender las funciones de cada componente el usuario tenga acotada la idea de la función completa de la interfaz gráfica de usuario.

De esta forma el usuario adquirirá los fundamentos necesarios para saber cómo navegar, cómo representar y qué representar, cómo pasar de un tipo de representación a otra, etc. para aprovechar al máximo las posibilidades que ofrece el programa

4.2.1. Ventana de inicio

Es la ventana a partir de la cual podemos explorar todas las funcionalidades del programa, introducción, representación, consulta, guardado y carga de los datos de la transmisión de la capa a analizar, de las características del material substrato y buffer, optimización de los datos y consulta de los resultados obtenidos tras la optimización.

Aquí seleccionaremos que modelos y funciones son los que queremos utilizar para la caracterización del material cargado en el programa.

Una vez se hayan realizado todos los pasos detallados en este manual, será desde la ventana de inicio desde donde podremos representar las gráficas siguientes gráficas:

- Índice de refracción del compuesto,

- Índice de absorción del compuesto
- Transmisión calculada
- Transmisión calculada y experimental

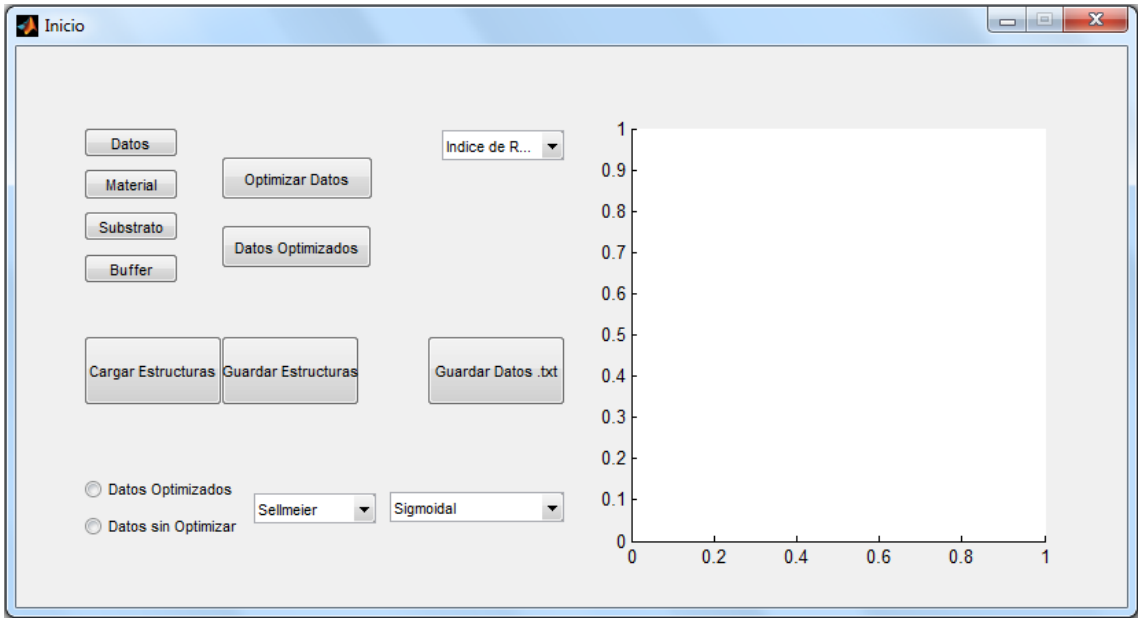


Figura 4.1 Ventana de Inicio del programa

4.2.2. Configuración de datos

En el botón "Datos" accedemos al menú donde podemos cargar los datos de la absorción de la capa que estemos llevando a estudio.

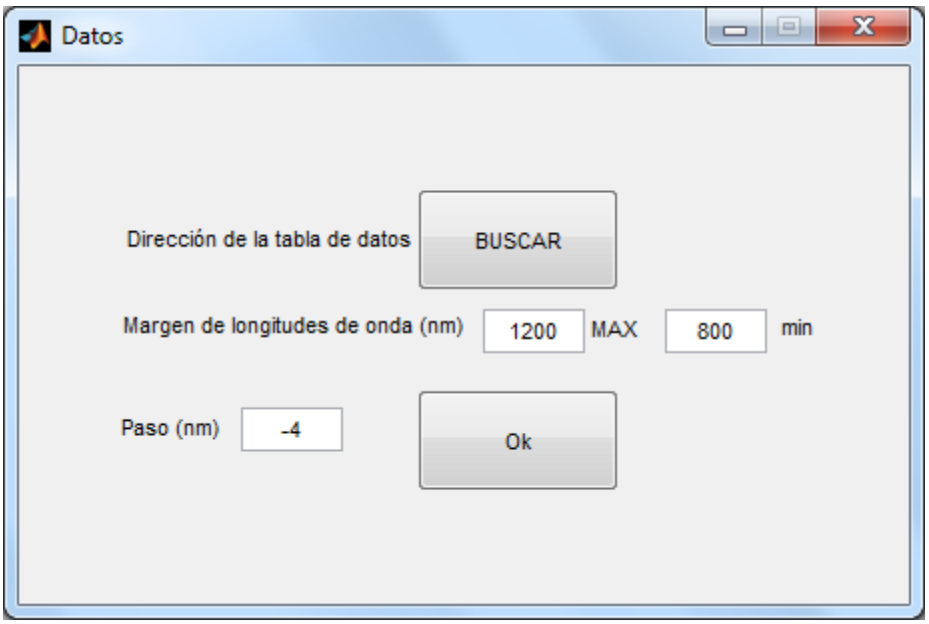


Figura 4.2 Introducción de los datos

En el botón “BUSCAR” se abrirá un menú que filtra la búsqueda por ficheros “.dat”, ficheros donde se almacenan los datos de la transmisión obtenidos mediante el ensayo con el espectrofotómetro., dicho fichero ha de ser un fichero de 2 columnas, la primera ha de representar los distintos valores de la longitud de onda y la segunda los valores de la transmisión en función de la longitud de onda.

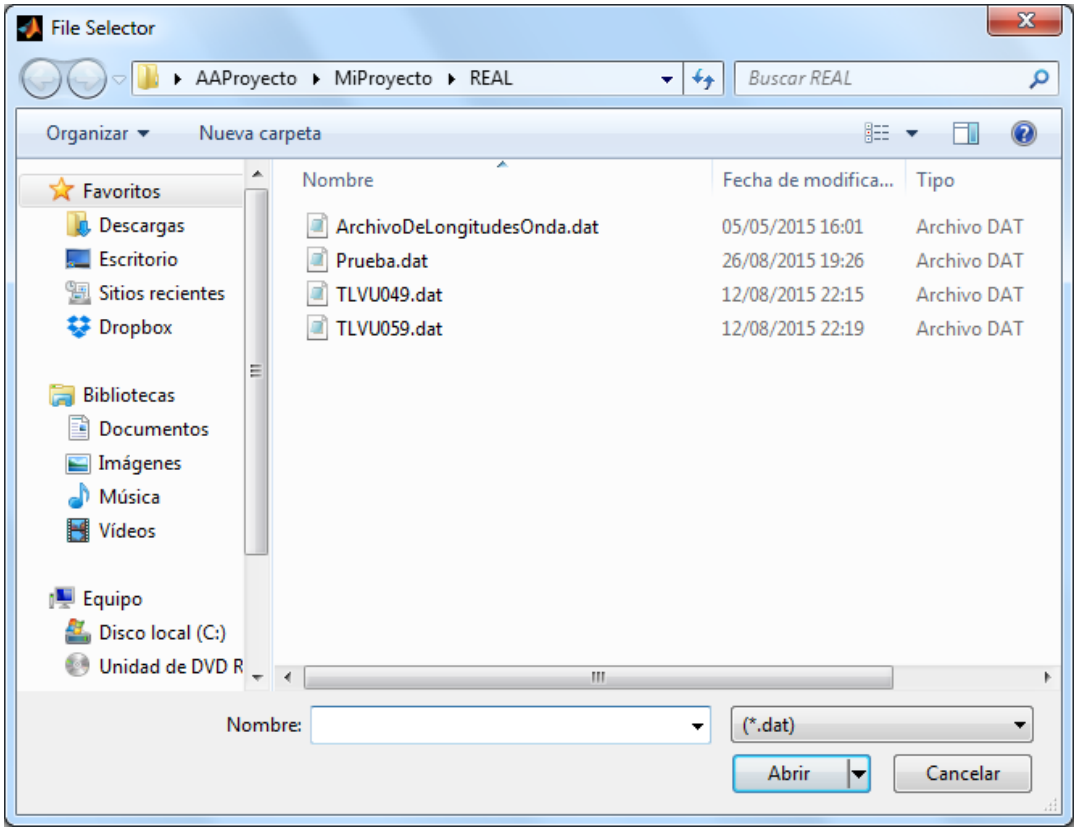


Figura 4.3 Selección de los datos de transmisión

En las casillas del Margen de longitudes de onda, introducimos los valores de longitud de onda, en nm, que queremos someter a estudio, por ejemplo (1.000, 600). Es este margen de longitudes de onda, en el que se realizarán las optimizaciones y ajustes de los datos y que posteriormente se representarán en las gráficas de la transmisión calculada.

En caso de haber realizado parte del proceso con un margen insuficiente o incorrecto, habrá que volver a la opción DATOS modificar el margen y realizar todo el proceso de nuevo, pues las ecuaciones y funciones llevadas a cabo hasta este punto, habrán sido realizadas con el margen erróneo.

Si el documento del que extraemos la tabla de datos está ordenado en sentido ascendente, para las longitudes de onda, debemos elegir un paso con signo positivo, si por el contrario las longitudes de onda están ordenadas en sentido descendente elegiremos un paso de signo negativo, por ejemplo (-4). Las unidades están recogidas en nanómetros, al igual que las del margen de longitudes de onda. Una vez introducidos estos datos pulsamos Ok y salimos. El programa nos devuelve a la ventana de Inicio, para poder seguir introduciendo la información necesaria.

4.2.3. Ventana de configuración del material

Ahora pulsamos en el botón "Material", donde accedemos al menú en el que modificaremos los valores de las propiedades del material, que queremos que nuestro programa optimice y ajuste, abriendo consigo la siguiente ventana mostrada en la (**figura 4.4)**

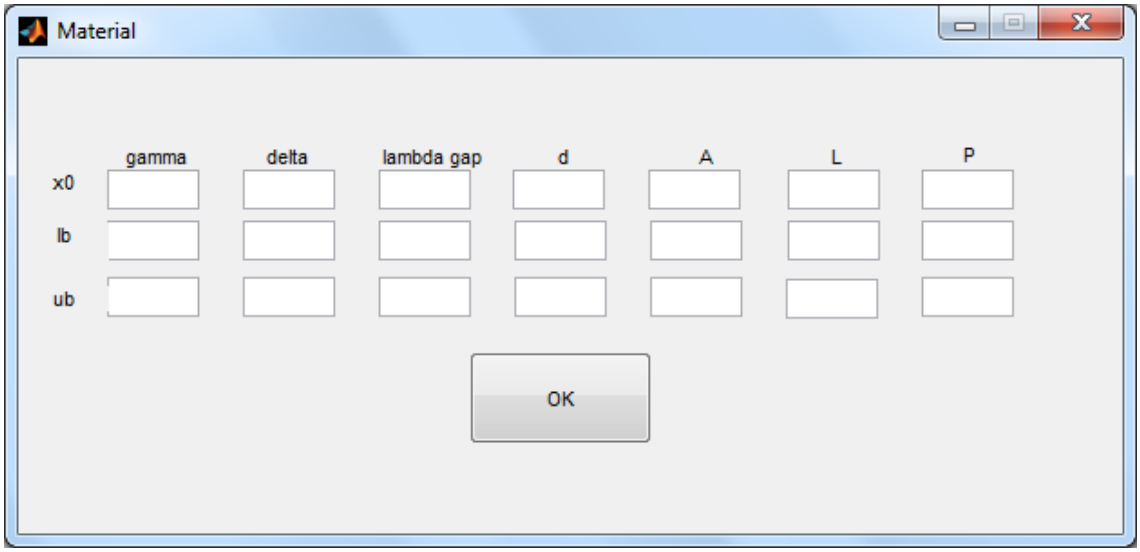


Figura 4.4 Introducción de los datos de la capa

Aunque a priori no conocemos los datos que integran esta opción de la interfaz, deberemos cumplimentarlos con unas características conocidas de un material similar, para que sirvan como base al programa para realizar las optimizaciones y ajustes correctamente, x0 representa al material, lb son los límites de optimización inferiores y ub son los límites de optimización superiores para cada uno de los parámetros de ajuste.

- Gamma
- Delta
- Lambda_gap
- D
- A

- L
- P

Este paso es importante y en gran medida de él dependerá el número de optimizaciones y ajustes que deberemos realizar y la fiabilidad de los resultados. A modo de ejemplo, los siguientes datos fueron utilizados para la optimización de una capa formada por aluminio e ITO.

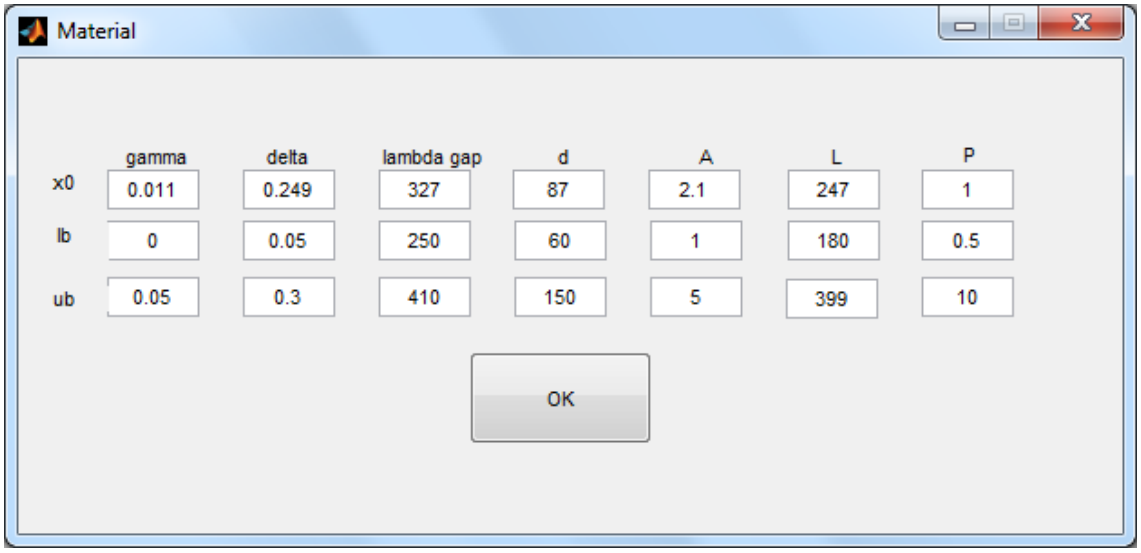


Figura 4.5 Datos de la capa introducidos para el ejemplo Al-ITO

En el caso de encontrarnos en este paso tras ya haber optimizado los datos y querer realizar una nueva optimización, debemos cumplimentar los datos de x0 copiando los datos que aparecen en la opción “Datos Optimizados” consiguiendo un ajuste que cada vez sea más fino.

Una vez introducidos los datos hacemos clic en OK y volvemos a la pantalla Inicio.

4.2.4. Ventana de configuración del substrato

En el botón Substrato accedemos al menú de configuración de las características del substrato en el que se ha depositado el material. Al hacerlo se nos desplegará un menú de dialogo en el que nos dará la opción de caracterizar el material con o sin buffer.

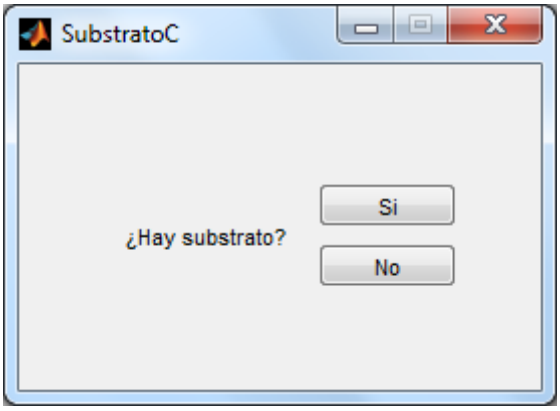


Figura 4.6 Selección del substrato

Este paso hay que realizarlo tanto si nuestro material no tiene substrato como sí si lo tuviera.

Pulsando "No" el programa asigna valor 0 al espesor del substrato y nos devuelve al menú de Inicio.

Cuando pulsemos en "Si" se abrirá la ventana "Substrato" dónde podremos introducir las características del substrato dónde se ha depositado nuestra muestra. En el ejemplo que exponemos en esta guía, el compuesto se depositó sobre vidrio, cuyas características son las siguientes:

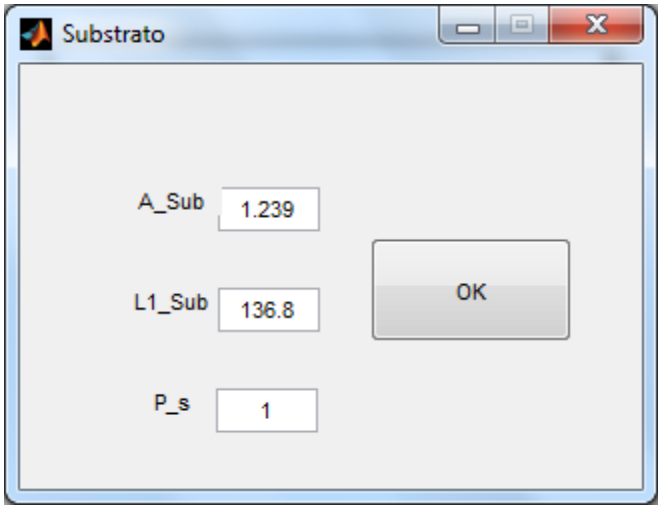


Figura 4.7 Introducción de los datos del substrato

- A_Sub
- L1_Sub
- P_s

Una vez introducidos los datos, pulsamos "Ok" y volvemos a Inicio

4.2.5. Ventana de configuración del buffer

El botón Buffer abre un cuadro de diálogo similar al que despliega la opción Substrato, este diálogo nos direccionará de igual manera a la pantalla de Inicio, en caso de no tener buffer y pulsar "No" y nos llevará a la opción Buffer en caso de pulsar en "Si". Igualmente es necesario indicar al programa que no hay buffer, para evitar problemas, sobre todo al cargar datos de otras aplicaciones.

En nuestro ejemplo en concreto, caracterizamos el material sin buffer. Asignando así, nuestro programa, valor 0 a todas las variables

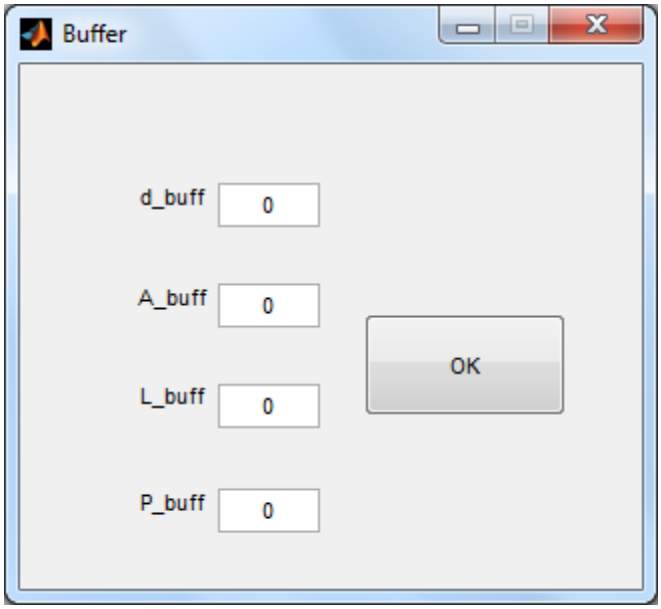


Figura 4.8 Introducción de los datos del buffer

- D_buff
- A_buff
- L_buff
- P_buff

4.2.6. Introducción de un nuevo índice de absorción o refracción

Una vez hayamos introducido todos los datos anteriores, procederemos a la optimización de dichos datos. Para ello primero debemos seleccionar que ecuaciones son las que queremos que el programa utilice para realizar la optimización "Sellmeier o Usuario" "Sigmoidal, Bergman, o Usuario" en ambos *pop-ups* la opción Usuario abre un desplegable para o bien crear una nueva ecuación de absorción

o refracción, dependiendo del caso concreto, o utilizar una ya creada y almacenada anteriormente.

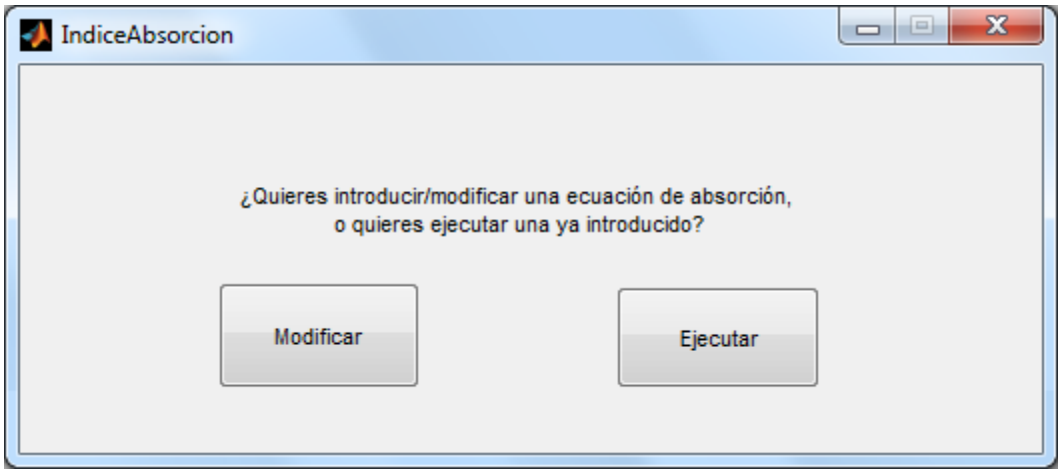


Figura 4.9 Selección de absorción

Al pulsar Modificar el programa nos direcciona a un archivo de MATLAB (.m), en el cual se deberá la ecuación de absorción que se quiera emplear

```
%Interfaz Gráfica                %
%Alejandro Villalba Sierra 2015 %
%Tutor Fernando Naranjo          %
%Departamento de electrónica UAH%
%Archivo donde introducir la ecuación de 2 o 3 variables que
determina%
%la absorción de la capa%

%Los parámetros de la capa vienen definidos por una estructura llamada
Representar:%
%Representar.AO%
%Representar.delta%
%Representar.lambda_gap%
%Representar.d%
%Representar.A%
%Representar.L%
%Representar.P%

%Un ejemplo utilizando la ecuación sigmoidal(la absorción, es el
parámetro "y"%
%que devuelve la función)%

%function y=f(x)
%global Representar Datos
%E=1239./Datos.Rango;
%Eg=1239/Representar.lambda_gap;
%deltaE=Representar.delta;
%y=Representar.AO./(1+(exp((Eg-E)./deltaE)));

%Un ejemplo utilizando la ecuación de Bergman
```

```
%y=Representar.AO+Representar.delta*exp(Representar.lambda_gap./Datos.
Rango);

function y=f(x)
global Representar Datos
y=Representar.AO+Representar.delta*exp(Representar.lambda_gap./Datos.R
ango);
```

En este fichero viene cargada por defecto la ecuación de Sellmeier, para realizar una prueba sencilla y ver que funciona correctamente. En los comentarios aparece también cómo hay que configurar la ecuación de Bergman para realizar la misma prueba.

En el caso de querer introducir una nueva ecuación, deberemos utilizar los campos de las estructuras que aparecen en las líneas de comentario. Es imprescindible nombrar estas estructuras como variables globales dentro de la función, puesto que en la GUI de MATLAB, es necesario nombrar dichas variables para poder utilizarlas en una función. Una vez hayamos definido la ecuación guardamos el archivo Absortion.m y el programa nos devolverá a la ventana de Inicio. Llegados a este punto, hay que recordar lo siguiente: Para utilizar la ecuación ahora almacenada en el fichero Absortion.m o Refraction.m deberemos volver a pulsar en la opción del pop-up "Usuario" y esta vez en el cuestionario elegir Ejecutar.

Es muy importante recordar este paso, porque el programa no correrá automáticamente el fragmento de código depositado en el fichero Absortion.m ni en el Refraction.m, por lo que siempre que queramos utilizar sus ecuaciones deberemos pulsar en Ejecutar.

4.2.7. Optimización de los datos

Bien, ahora para realizar la optimización debemos seleccionar en los pop-up de la ventana "Inicio" de nuevo las funciones que queremos utilizar y en el botón "Optimizar Datos" donde indicamos al programa cuales son los atributos con los que queremos que optimice la función "lsqnonlin" esta función ajusta nuestros datos base almacenados previamente en "Material" mediante aproximación por mínimos cuadrados. Este proceso puede resultar reiterativo, pero ha sido programado de esta manera para recordar al usuario las funciones que está utilizando durante todo el proceso, además de permitir cambiar los modelos utilizados cuando sea necesario, sin tener que repetir el proceso desde el principio.

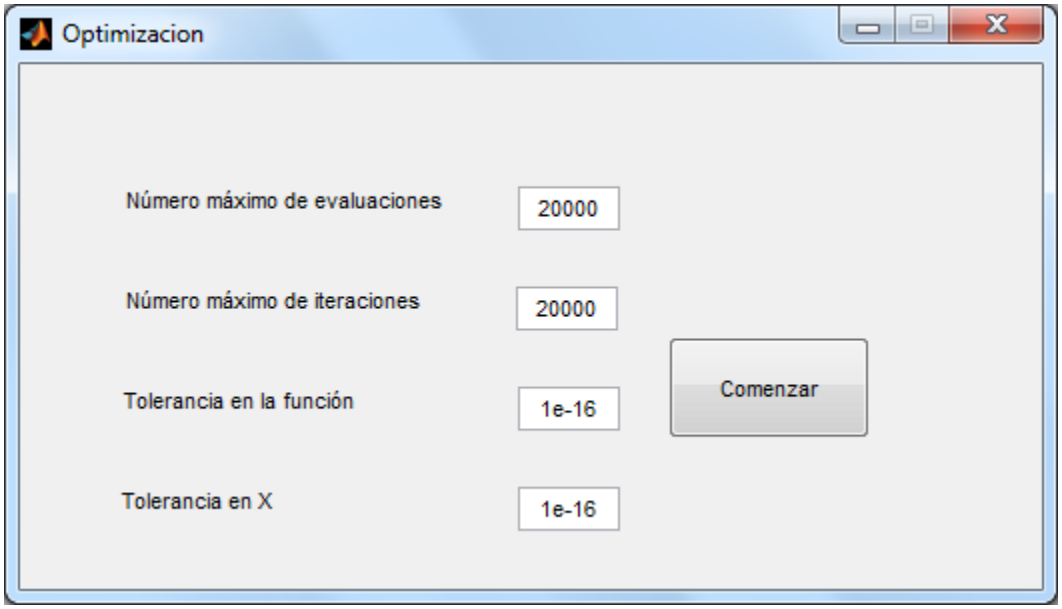


Figura 4.10 Ventana de optimización

La interfaz siempre nos indicará los valores para los atributos de la función, por defecto, que aparecen arriba. Que como se indica hacen referencia al número máximo de evaluaciones que realizará la función, el número máximo de iteraciones, la tolerancia en el valor devuelto por la función y la tolerancia en el valor de la componente evaluada.

La optimización puede llevar un par de minutos. En el momento que acabe se cerrará automáticamente la ventana de "Optimización" devolviéndonos al menú de "Inicio".

4.2.8. Representación y corrección de los datos obtenidos

Con los datos optimizados sólo queda representarlos, para ello elegimos de nuevo las ecuaciones que hemos seleccionado en la optimización "Sellmeier o Usuario" "Sigmoidal, Bergman, o Usuario" y en el Pop-up de la gráfica elegimos qué gráficas queremos representar:

- Índice de refracción.
- Absorción.
- Transmisión calculada.
- Transmisión calculada y experimental.

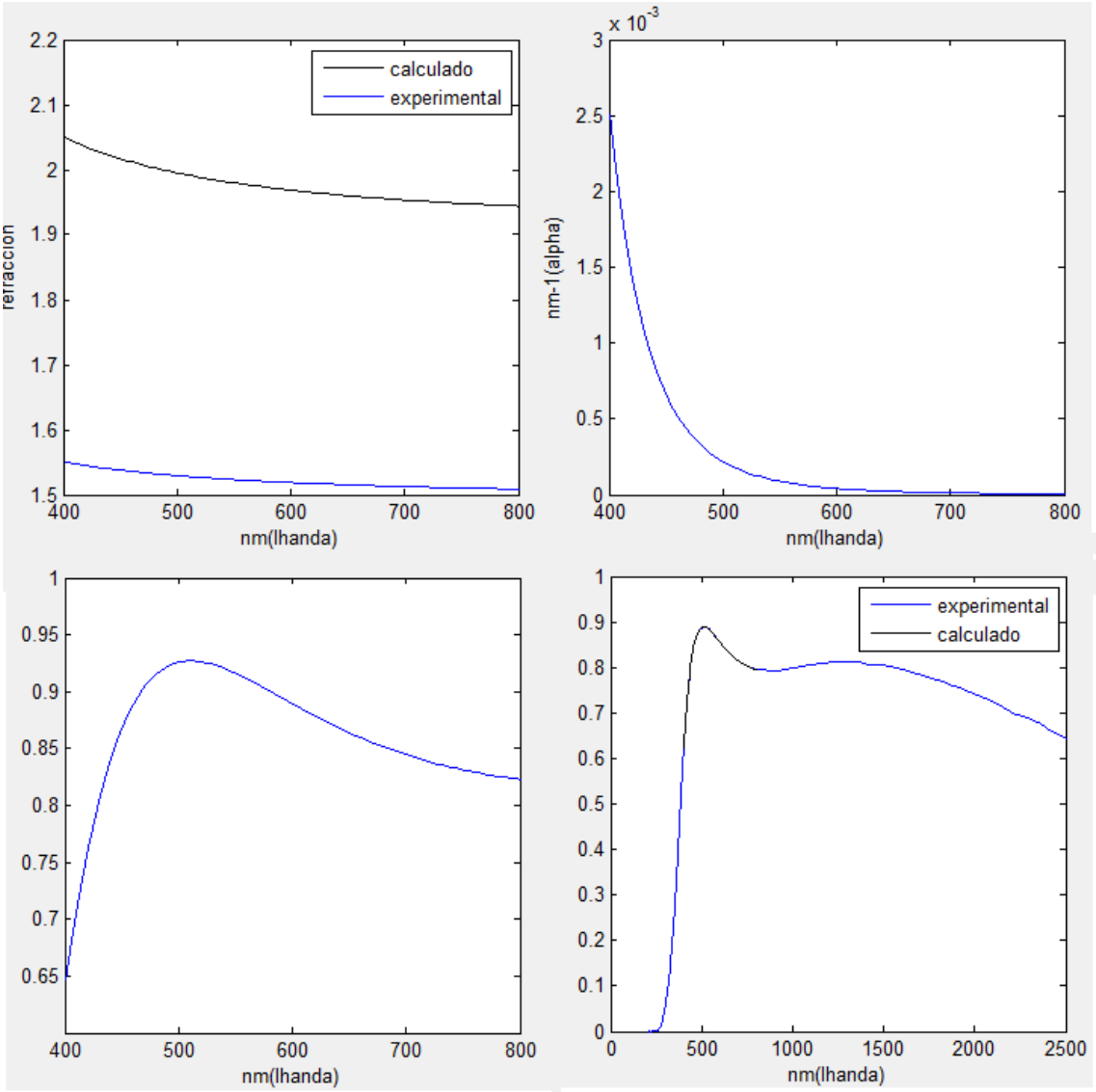


Figura 4.11 Representación de las gráficas de absorción, refracción y transmisiones

En caso de que el ajuste que haya realizado el programa no sea satisfactorio, podremos acceder a los datos optimizados en el botón con su mismo nombre y allí ver y modificar los datos para que el ajuste sea de una mayor calidad.

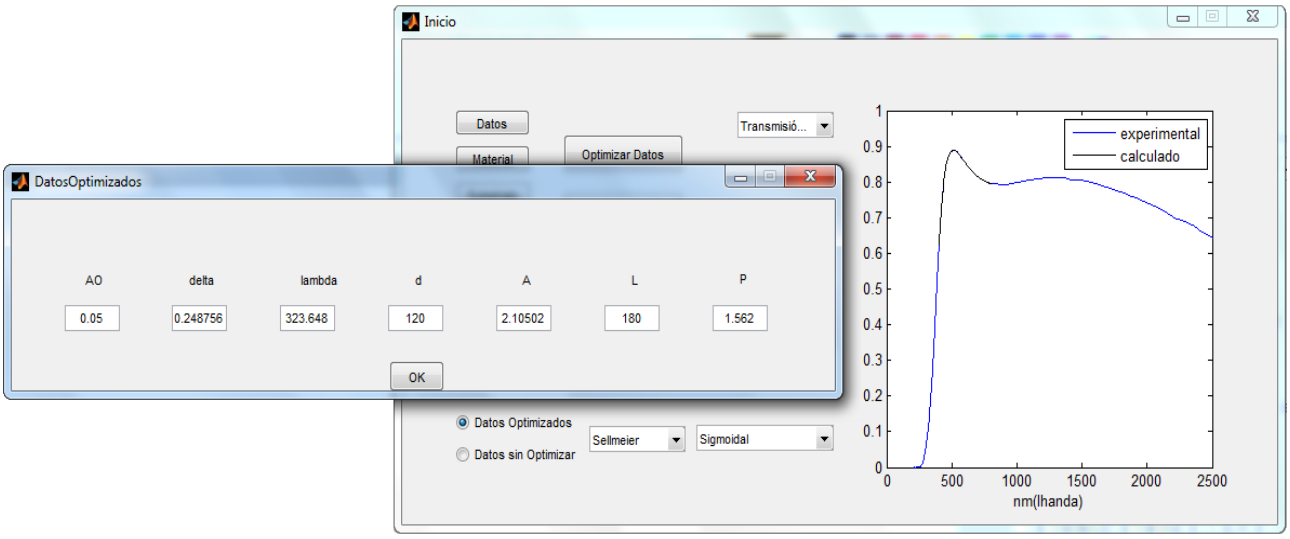


Figura 4.12 Representación de los datos optimizados, en gráfica con sus parámetros

El procedimiento a seguir en esta sección es el siguiente:

1. Modificar los datos en la ventana DatosOptimizados.

2. Ir a la ventana de Inicio y seleccionar en el selector Datos Optimizados, pues son estos los que hemos modificado en el paso anterior.

3. Elegir en el *pop-up* la gráfica Transmisión calculada o la gráfica Transmisión calculada y experimental, pues es la transmisión calculada la que estamos modificando al variar los parámetros en DatosOptimizados .

4. Una vez hayamos mejorado la proximidad entre ambas representaciones (Transmisión calculada y Transmisión experimental). Nos introduciremos en el menú Material, dónde copiaremos los datos que hemos modificado en DatosOptimizados, modificando en consecuencia los límites de optimización.

5. Optimizamos de nuevo.

Este proceso se puede realizar tantas veces como se necesite.

Si hemos obtenido un resultado satisfactorio y quisiéramos guardar las estructuras que hemos introducido y obtenido a lo largo de todo el proceso, para poder acceder a ellas en otro momento, haríamos *click* en el botón con su mismo nombre “Guardar Estructuras” y quedarían

perfectamente guardadas para posteriores visualizaciones. El programa establece previamente el tipo de fichero .mat.

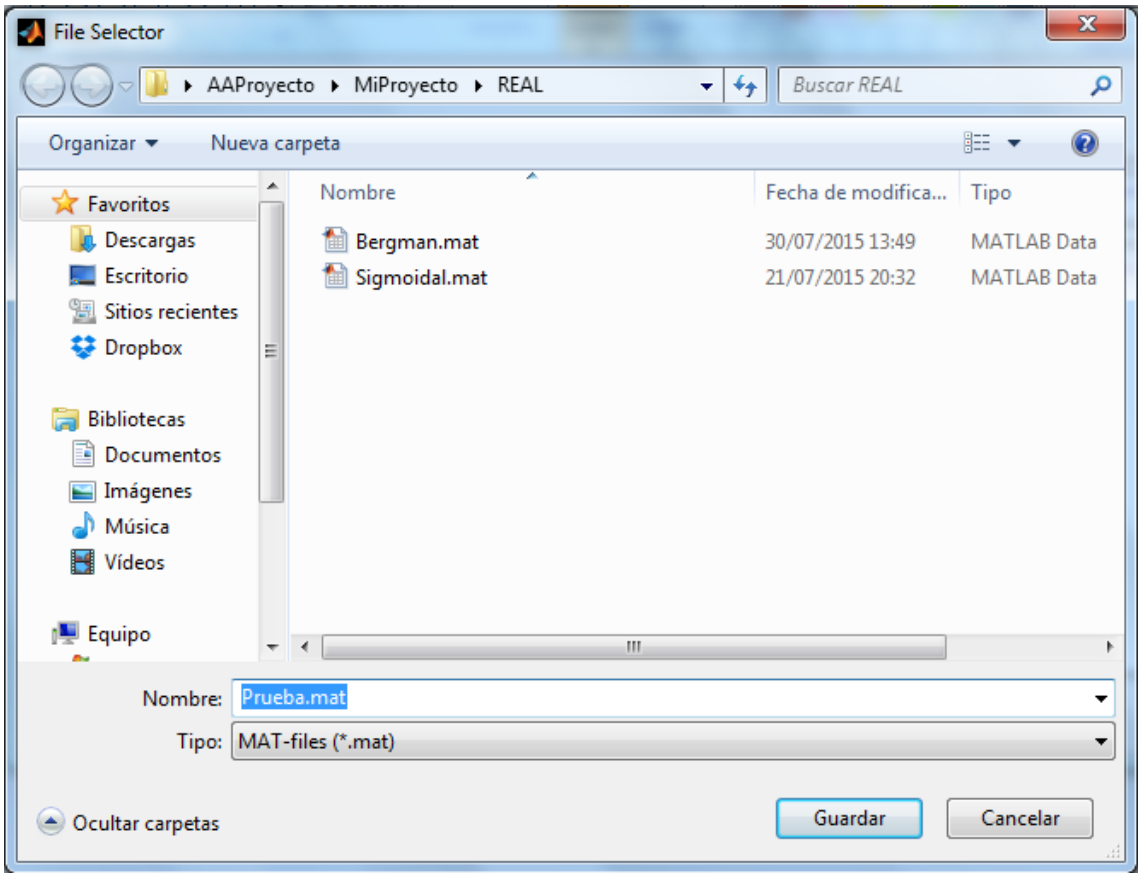


Figura 4.13 Guardado de estructuras .mat

En el caso de que queremos cargar unas estructuras guardadas en otras ejecuciones del programa, como las que aparecen en la imagen de arriba, pulsamos en el botón “Cargar Estructuras”. Esta funcionalidad carga todas las estructuras necesarias para una correcta ejecución del programa, por lo que podremos consultar todos los datos que introdujimos así como las gráficas y resultados que de ellos se obtuvieron.

Por último, en caso de querer obtener un fichero (.txt) en el que aparezcan los resultados con los que el programa realiza las gráficas, hacemos clic en el botón “Guardar Datos .txt” y nos aparecerá la siguiente ventana.

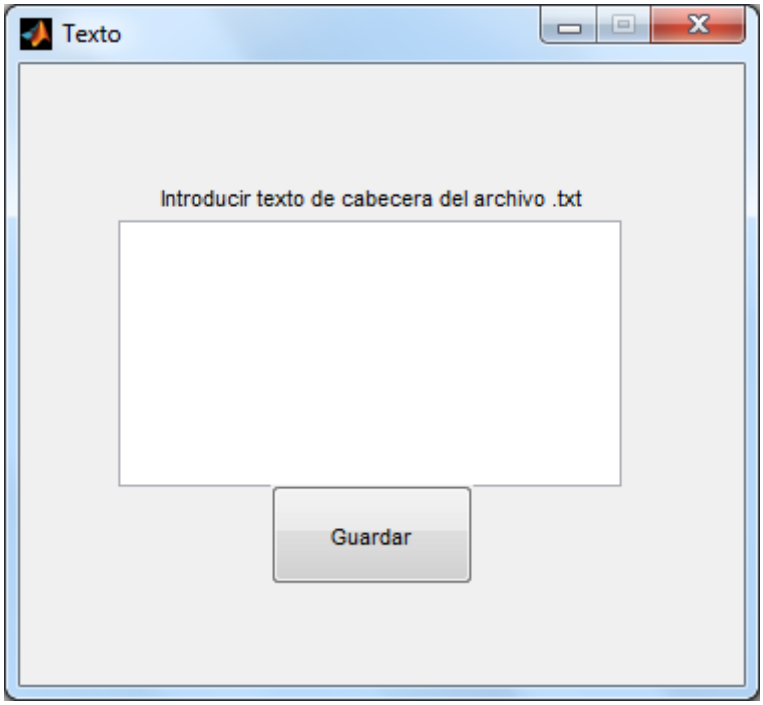


Figura 4.14 Ventana para el guardado del archivo .txt

Este menú puede rellenarse con la información que se considere necesaria, el texto que escribamos en el recuadro blanco, aparecerá como cabecera del fichero (.txt) que se genera, de manera que podamos aportar información valiosa para almacenar los datos de una forma comprensible para futuras consultas, como por ejemplo:

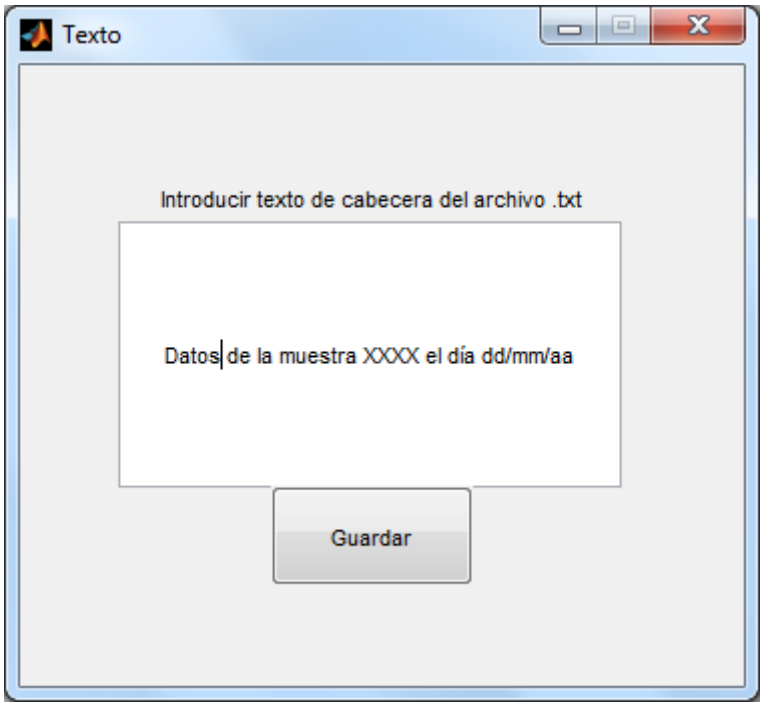
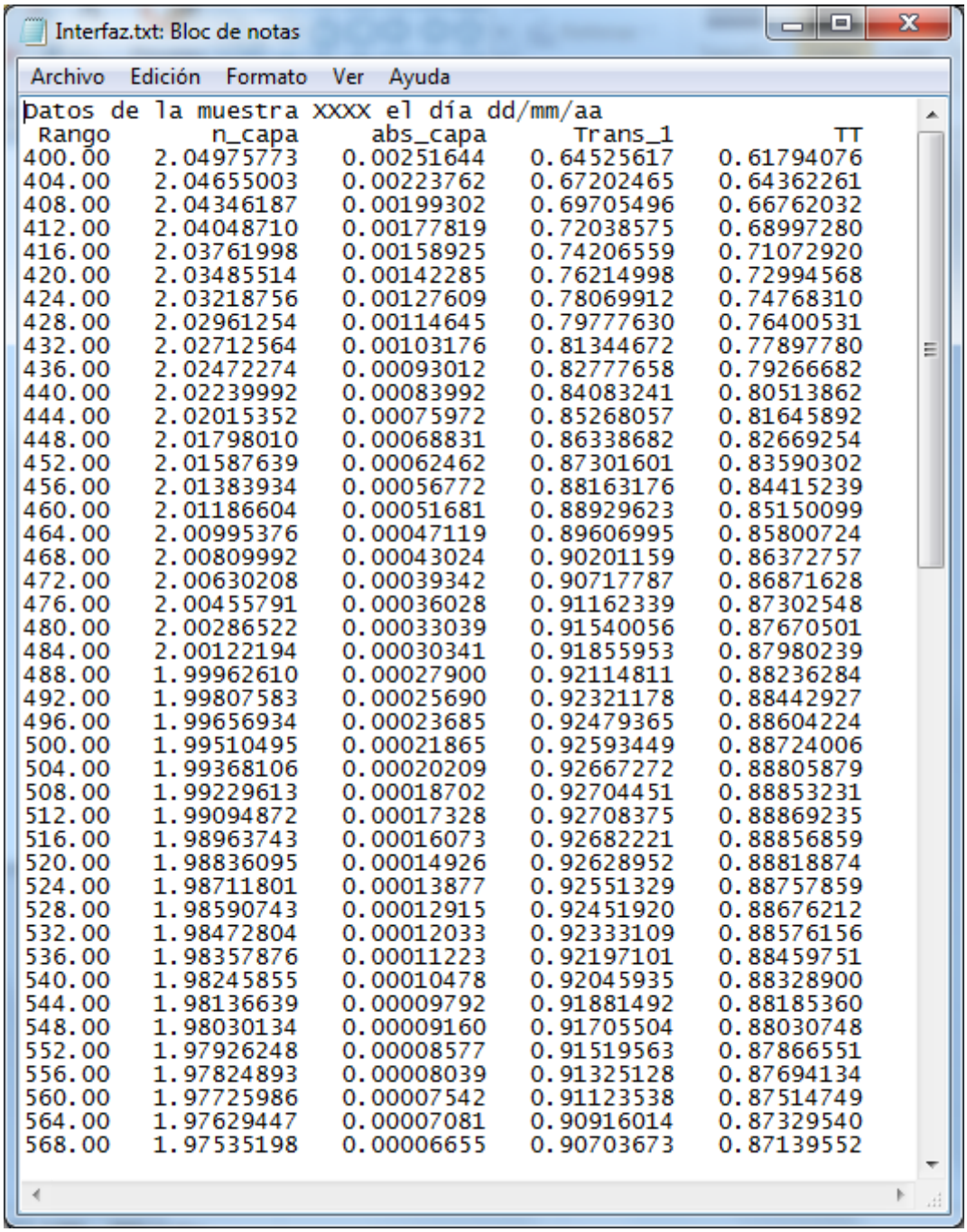


Figura 4.15 Ventana para el guardado del archivo .txt

De este modo obtendremos un fichero de texto en el que aparecerán la refracción y la absorción de la capa, así como la transmisión experimental (Trans_1) y la transmisión calculada (TT) todos ellos ordenados dentro del margen de longitudes de onda que indicamos en el primer paso



Datos de la muestra XXXX el día dd/mm/aa

Rango	n_capa	abs_capa	Trans_1	TT
400.00	2.04975773	0.00251644	0.64525617	0.61794076
404.00	2.04655003	0.00223762	0.67202465	0.64362261
408.00	2.04346187	0.00199302	0.69705496	0.66762032
412.00	2.04048710	0.00177819	0.72038575	0.68997280
416.00	2.03761998	0.00158925	0.74206559	0.71072920
420.00	2.03485514	0.00142285	0.76214998	0.72994568
424.00	2.03218756	0.00127609	0.78069912	0.74768310
428.00	2.02961254	0.00114645	0.79777630	0.76400531
432.00	2.02712564	0.00103176	0.81344672	0.77897780
436.00	2.02472274	0.00093012	0.82777658	0.79266682
440.00	2.02239992	0.00083992	0.84083241	0.80513862
444.00	2.02015352	0.00075972	0.85268057	0.81645892
448.00	2.01798010	0.00068831	0.86338682	0.82669254
452.00	2.01587639	0.00062462	0.87301601	0.83590302
456.00	2.01383934	0.00056772	0.88163176	0.84415239
460.00	2.01186604	0.00051681	0.88929623	0.85150099
464.00	2.00995376	0.00047119	0.89606995	0.85800724
468.00	2.00809992	0.00043024	0.90201159	0.86372757
472.00	2.00630208	0.00039342	0.90717787	0.86871628
476.00	2.00455791	0.00036028	0.91162339	0.87302548
480.00	2.00286522	0.00033039	0.91540056	0.87670501
484.00	2.00122194	0.00030341	0.91855953	0.87980239
488.00	1.99962610	0.00027900	0.92114811	0.88236284
492.00	1.99807583	0.00025690	0.92321178	0.88442927
496.00	1.99656934	0.00023685	0.92479365	0.88604224
500.00	1.99510495	0.00021865	0.92593449	0.88724006
504.00	1.99368106	0.00020209	0.92667272	0.88805879
508.00	1.99229613	0.00018702	0.92704451	0.88853231
512.00	1.99094872	0.00017328	0.92708375	0.88869235
516.00	1.98963743	0.00016073	0.92682221	0.88856859
520.00	1.98836095	0.00014926	0.92628952	0.88818874
524.00	1.98711801	0.00013877	0.92551329	0.88757859
528.00	1.98590743	0.00012915	0.92451920	0.88676212
532.00	1.98472804	0.00012033	0.92333109	0.88576156
536.00	1.98357876	0.00011223	0.92197101	0.88459751
540.00	1.98245855	0.00010478	0.92045935	0.88328900
544.00	1.98136639	0.00009792	0.91881492	0.88185360
548.00	1.98030134	0.00009160	0.91705504	0.88030748
552.00	1.97926248	0.00008577	0.91519563	0.87866551
556.00	1.97824893	0.00008039	0.91325128	0.87694134
560.00	1.97725986	0.00007542	0.91123538	0.87514749
564.00	1.97629447	0.00007081	0.90916014	0.87329540
568.00	1.97535198	0.00006655	0.90703673	0.87139552

Figura 4.16 Formato del archivo .txt

5. BIBLIOGRAFÍA

- *Nakamura, S. (1997). Análisis numérico y visualización gráfica con MATLAB.*

- *Lorenzo, E. (2014). Ingeniería fotovoltaica. Volumen III*

- *Carrera Amuriza, A.(2004). Introducción a Matlab y a la creación de interfaces gráficas*

- *http://es.mathworks.com/help/matlab/*