UNIVERSIDAD COMPLUTENSE MADRID

Proyecto de Innovación y Mejora de la Calidad Docente

Convocatoria 2014

Nº de proyecto: 200

Título: **Desarrollo de herramientas interactivas con acceso remoto para el aprendizaje de óptica biomédica**

Responsable: **Eduardo Cabrera Granado**

Centro: Facultad de Óptica y Optometría

Departamento: Óptica

1. **Objetivos propuestos en la presentación del proyecto** (Máximo 2 folios)

Los objetivos propuestos en el documento de solicitud del proyecto eran los siguientes:

1. Generación de tutoriales de uso de los documentos y de las herramientas de cálculo numérico útiles para el estudio de los datos típicos obtenidos en experimentos de Óptica Biomédica. Estos tutoriales se realizarán en formato vídeo (screencasts) así como en HTML y en el propio formato de los documentos.

2. Generación de documentos interactivos con texto, aplicaciones multimedia y programas de cálculo, que permitan una comprensión más profunda de los conceptos explicados en clase. También se elaborará material de apoyo para explicar los conceptos y metodología utilizada en las prácticas, así como proporcionar modelos base para realizar el tratamiento de datos necesario para obtener los resultados.

3. Puesta en marcha de un servidor al que los estudiantes puedan acceder para consultar y modificar los documentos generados, así como ejecutar los programas y simulaciones presentes en ellos utilizando diferentes parámetros y casos de estudio.

4. Difusión de los resultados del proyecto en un congreso de docencia.

5. Implantación efectiva en el curso 2014-2015 en la asignatura de Óptica Biomédica.

Estos cinco objetivos se enmarcan dentro de una aspiración mayor: la de proveer a los estudiantes de una herramienta flexible que permita la consulta de contenidos en distintos formatos (texto, vídeo, simulación), el análisis de datos proporcionados o medidos por parte del estudiante en el laboratorio, y la simulación de modelos computacionales que permitan entender los procesos biológicos estudiados en la asignatura de Óptica Biomédica.

2. **Objetivos alcanzados** (Máximo 2 folios)

Los objetivos propuestos han sido alcanzados en un porcentaje muy elevado. En aquellos en los que la financiación solicitada era indispensable para llevarlos a cabo, se han estudiado y puesto en marcha alternativas que no requerían de esa financiación. El objetivo de difundir los resultados del proyecto en un congreso de docencia no ha podido ser satisfecho, dada la ausencia de financiación del proyecto.

El grado de alcance de los objetivos enumerados en el punto anterior se detalla a continuación.

Objetivos	1	2	3	4	5
Grado de alcance	100%	80%	75%	---	100%

Tabla 1: Grado de alcance de los objetivos.

1. El primer objetivo consistía en generar tutoríales de uso de este tipo de documentos en formato HTML, vídeo y en el formato propio de los documentos interactivos.
Durante la ejecución del proyecto se han elaborado dos documentos de referencia para el uso de IPython Notebook así como para aprender a utilizar el lenguaje de programación Python como herramienta de cálculo numérico y de análisis de datos.

En el primero de estos documentos se abordan tanto la estructura como las operaciones básicas de control de los contenidos de IPython Notebook. En el segundo de estos tutoriales se profundiza en la realización de tareas fundamentales en el análisis de resultados en un laboratorio como son la carga de datos desde ficheros, cálculo de resultados estadísticos básicos, generación y personalización de figuras, y por último, realización de ajustes lineales y no lineales de datos experimentales a modelos teóricos.

Además de estos documentos, se ha realizado un vídeo explicativo sobre el uso del servicio SageMath Cloud, que ha sido ampliamente utilizado durante el desarrollo del proyecto por parte de profesores y estudiantes, sustituyendo así al servidor propio (veáse el punto 3 de esta misma sección).

Este material generado por los profesores de la asignatura cubre los distintos apartados planteados dentro de este objetivo en la solicitud del proyecto.

2. La base del proyecto era la generación de material de trabajo en el formato de IPython Notebook para cubrir distintos aspectos de la asignatura. El trabajo de los profesores participantes en este proyecto ha dado lugar a 11 documentos:

- Dos documentos de introducción al manejo de IPython Notebook y a las funciones básicas de análisis de datos: cargar datos desde un fichero, visualización en figuras y ajuste de los datos a modelos teóricos. Estos documentos han formado parte de una primera práctica numérica y han estado a disposición de los estudiantes de forma permanente para su consulta posterior a lo largo del curso.
- Un documento interactivo de explicación de los conceptos más importantes de la propagación de la radiación en tejidos biológicos.
- Un documento de análisis de los datos medidos en el laboratorio en la práctica de espectroscopía de medios biológicos.

- Un documento de trabajo con la explicación de los conceptos más importantes de la espectroscopía de reflectancia difusa.
- Un documento de análisis de los resultados de la práctica de espectroscopía de reflectancia difusa.
- Un guión de prácticas, explicando la metodología a utilizar y los conceptos fundamentales de la práctica de microscopía de fluorescencia.
- Dos documentos de trabajo, previo a la sesión de laboratorio y posterior de análisis de los resultados de la práctica de microscopía de fluorescencia.
- Un documento de introducción teórica a los conceptos de la Teoría Difraccional de la Imagen.
- Dos documentos asociados a una práctica numérica de análisis de la Teoría Difraccional de la Imagen. En el primero de ellos el estudiante analiza la descripción de la aberración de onda mediante el desarrollo en polinomios de Zernike, mientras que en el segundo se explora el efecto de estas aberraciones en las funciones características de un sistema óptico, la Función de Dispersión de punto y la Función de Transferencia de Modulación.

Dada la favorable implantación de este tipo de recursos en la asignatura, se prevé elaborar más material el próximo curso, especialmente de carácter teórico, como complemento y apoyo a los conceptos explicados en clase.

3. La puesta en marcha de un servidor propio para dar acceso a los estudiantes en la consulta y trabajo con los documentos elaborados no ha sido posible debido a la falta de financiación. Sin embargo, se ha encontrado como alternativa el uso del servicio gratuito SageMath Cloud, facilitado por la Universidad de Washington y desarrollado por el profesor William Stein de dicha Universidad. Este servicio proporciona una plataforma para poder ejecutar documentos IPython Notebook así como almacenar archivos, previsualizar PDFs, edición de textos y trabajo con otras herramientas de cálculo como es Sage. Además, se ha aprovechado el desarrollo dentro de SageMath Cloud de una herramienta de gestión de cursos, facilitando la asignación de tareas a los estudiantes, recogida de trabajos y calificación. Siendo tan importante en el desarrollo del curso, se ha considerado importante la elaboración de una guía en vídeo de uso básico de esta herramienta online. Este material se ha añadido como resultado del presente proyecto.
La alternativa encontrada, aunque no iguala la flexibilidad y control de un servidor propio como se había propuesto inicialmente, satisface el objetivo propuesto en este punto de facilitar a los estudiantes un servicio de acceso remoto que les libere de la instalación de software en sus equipos personales, presentando además algunas ventajas comentadas como la gestión de cursos y la capacidad de trabajar con otro tipo de archivos.

4. Este objetivo no se ha planteado en la realización del proyecto debido a la falta de financiación.

5. Todo el trabajo realizado ha sido implantado en la docencia del presente curso 2014-2015 en la asignatura de Óptica Biomédica cumpliendo por tanto este objetivo de forma completa.

3. **Metodología empleada en el proyecto** (Máximo 1 folio)

Tanto para el trabajo de los estudiantes con los documentos interactivos como para la coordinación y desarrollo colaborativo de los mismos, se ha utilizado intensamente la plataforma gratuita SageMath Cloud de la Universidad de Washington. Es por ello que en primer lugar los profesores participantes han debido familiarizarse con las posibilidades de este servicio online así como probar sus limitaciones y posibles problemas de cara a su uso

en la impartición de la asignatura Óptica Biomédica en el presente curso 2014-2015.

Dadas las ventajas de esta plataforma (edición colaborativa con sincronización de cambios entre distintas cuentas, gestión de colaboradores, herramienta de creación de cursos para asignar y recoger tareas), y a la falta de poder instalar un servidor propio que permitiese el acceso remoto a los documentos, se decidió utilizar este servicio como canal para dejar a disposición de los estudiantes los documentos, monitorizar sus progresos, recoger las tareas asignadas y por último, calificar los trabajos realizados.

Por otro lado, se han elaborado distintos documentos de trabajo para los estudiantes cubriendo los módulos que componen la asignatura de Óptica Biomédica. En todos estos documentos se ha intentado que el estudiante no sólo comprenda de una manera más efectiva los conceptos tratados, sino que también adquiera habilidades en el manejo de datos y análisis por medio de un programa de cálculo numérico. En este sentido, se ha promovido el uso de funciones y comandos lo más general posible, de modo que el paso entre distintos programas (Ipython y MATLAB/Octave, fundamentalmente) sea sencillo en un futuro. Además, estos documentos han sido accesibles a los estudiantes en su cuenta de SageMath Cloud hasta final de curso, permitiéndoles profundizar en los distintos temas dada la libertad para modificar los casos planteados y el conocimiento del código empleado desde principio de curso:

a) Módulo de interacción radiación-materia: Se ha elaborado un documento de explicación de la propagación de radiación en tejidos biológicos, así como un documento de análisis de los datos obtenidos en la práctica asociada a los temas tratados en este apartado de la asignatura, incluyendo la obtención de absorbancias de muestras problemas o visualización y análisis de espectros de transmisión, entre otros ejercicios.

b) Módulo de bio-espectroscopía: Se han desarrollado dos documentos de trabajo para la práctica de espectroscopía de reflectancia difusa, incluyendo por ejemplo el ajuste por parte de los estudiantes de los datos medidos a modelos teóricos (explicados en el propio documento) con el fin de extraer información como puede ser el nivel de oxigenación en sangre. Además, se han generado tres documentos de trabajo dedicados a la práctica de microscopía de fluorescencia, incluyendo el análisis de imágenes captadas por el microscopio construido por cada estudiante con el fin de obtener información sobre el tamaño de estructuras.

c) Módulo de bio-imagen: Se ha elaborado un documento introductorio a los conceptos de la Teoría Difraccional de la Imagen, así como dos documentos de trabajo que han compuesto una práctica numérica sobre este tema. En estos documentos, el estudiante ha podido explorar con libertad el efecto de distintas aberraciones en la formación de la imagen, calcular métricas de calidad de imagen, y visualizar funciones características de un sistema óptico.

4. **Recursos humanos** (Máximo 1 folio)

El proyecto no contemplaba recurrir a expertos externos, llevándose a cabo por tanto el trabajo por parte de los profesores participantes. El desarrollo del proyecto se ha efectuado de una manera coordinada, indispensable dada la implementación inmediata de los resultados del proyecto en el curso docente 2014-2015. En este sentido, el uso de SageMath Cloud como plataforma, no solo de trabajo para los estudiantes, sino de desarrollo colaborativo de los distintos documentos ha sido de gran ayuda.

Dado que la mayoría de los documentos generados han sido utilizados en el laboratorio de la asignatura, todos los profesores que imparten docencia en ella han colaborado en su desarrollo, bien con la implementación de código, bien con indicaciones sobre los elementos a tratar en cada documento. Estos profesores han sido Miguel Ángel Antón Revilla (quien también ha elaborado un documento de desarrollo teórico sobre la propagación de radiación en medios biológicos), Sonia Melle Hérnandez, José Manuel López Alonso y Eduardo Cabrera Granado. En cuanto a los profesores participantes cuya docencia no está asociada a la asignatura de Óptica Biomédica, el profesor Oscar Gómez Calderón ha desarrollado activamente varios de los documentos utilizados en las prácticas. Por otro lado, la profesora Elena Díaz García ha participado en la generación de los documentos de introducción a Python y al manejo de IPython Notebook. Tmbién ha elaborado, junto a Francisco Arrieta Yañez, el vídeo explicativo de primeros pasos en la plataforma SageMath Cloud. Por su parte, el profesor Fernando Carreño Sánchez ha participado activamente en el desarrollo de los documentos de trabajo para el tema de Teoría Difraccional de la Imagen.

5. **Desarrollo de las actividades** (Máximo 3 folios)

El cronograma planteado en la solicitud del presente proyecto preveía la puesta en marcha del servidor de acceso remoto para los estudiantes justo antes del comienzo de las clases. Dada la imposibilidad de tener este servidor propio, se adaptó el desarrollo de las tareas del proyecto para buscar en primer lugar una alternativa para salvar la necesidad de instalación del programa IPython (necesario para visualizar los documentos interactivos en formato IPython Notebook) por parte de los estudiantes, lo cual puede ser una barrera inicial, en algunos casos importantes. Por ello el desarrollo cronológico de las actividades ha sido el siguiente:

1. Junio- Septiembre: Durante este periodo la actividad fundamental desarrollada consistió en la busqueda de alternativas al servidor propio. Descartada la opción de requerir una instalación en los equipos personales de los estudiantes, nos centramos en dos servicios web que ofrecen la posibilidad de ejecutar documentos en formato IPython Notebook: Wakari y SageMath Cloud. Aunque ambas opciones son adecuadas para trabajar con los documentos interactivos, SageMath Cloud añade la posibilidad de editar otro tipo de ficheros (LaTeX, texto, etc), provee de una mayor capacidad de almacenamiento y memoria RAM, y por último, permite la gestión de cursos facilitando la asignación y entrega de tareas masiva, pudiendo enviar a los estudiantes los ficheros necesarios para llevarlas a cabo en su cuenta con sólo presionar un botón.

Una vez elegido el servicio, el equipo de profesores puso a prueba su capacidad y facilidad de manejo antes de tomar la decisión final de utilizarlo como plataforma para el trabajo en la asignatura.

2. Septiembre-Octubre: En el mes de Septiembre y comienzo del mes de Octubre se elaboraron los tutoriales para el uso de IPython Notebook y de Python para cálculo científico. Se elaboró una práctica de la asignatura dedicada a aprender a manejar este lenguaje en

tareas como la carga de ficheros, visualización de figuras y ajuste a distintos modelos teóricos de los datos experimentales. De este modo, los estudiantes pudieron trabajar junto al profesor en los problemas que se les presentaban y aprendían habilidades fundamentales en el trabajo de laboratorio que serían requeridas posteriormente. Estos tutoriales han estado accesibles durante todo el curso en la cuenta de cada estudiante para facilitar su consulta ante cualquier duda.

3. Octubre- Enero (2015): Durante este periodo se ha elaborado el grueso de los documentos interactivos, tanto de trabajo para las prácticas de la asignatura como de apoyo a los conceptos teóricos explicados. En este sentido cabe destacar la edición de los documentos de un modo colaborativo a través del servicio SageMath Cloud, lo que ha facilitado el desarrollo de esta actividad.

También se han llevado a cabo varias reuniones de coordinación entre los profesores participantes para perfilar y criticar las distintas propuestas de documentos interactivos elaborados, así como sugerir nuevos documentos en base al desarrollo del curso.

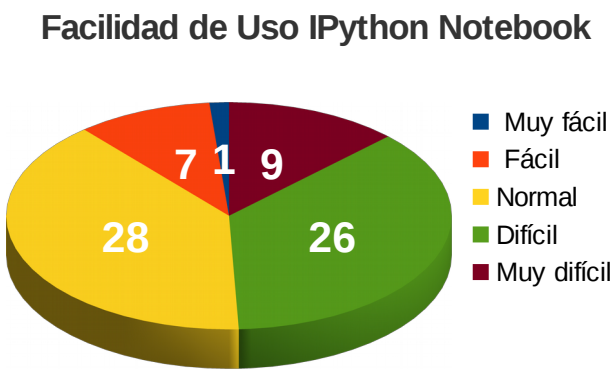
4. Enero (2015). Desde la última semana del mes de Enero de 2015 los estudiantes han podido contestar a una encuesta anónima planteada en el espacio de la asignatura dentro del Campus Virtual de la UCM. Los resultados de esta encuesta se pueden consultar en el apartado de Anexos.

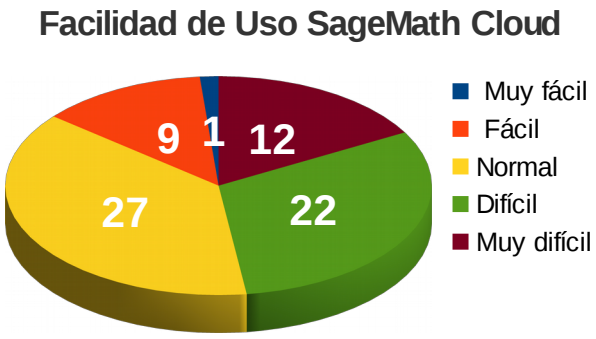
Durante el trabajo con los documentos en la plataforma SageMath Cloud los estudiantes han experimentado caídas de conexión e intentos fallidos de salvar los cambios efectuados en los documentos. Estos problemas se han dado fundamentalmente durante el último mes del primer cuatrimestre, coincidiendo con el desarrollo de la última práctica del laboratorio de la asignatura. Aunque estas incidencias parece que ya están subsanadas (según se ha anunciado en los canales de consulta con los desarrolladores de SageMath Cloud), sí han provocado retrasos en la ejecución de esa práctica e incertidumbre a la hora de guardar los ejercicios contestados. Estos problemas podrían explicar (en parte) la disparidad en la valoración de este servicio que se manifiesta en los resultados de la encuesta. La adquisición de material informático suficiente para poner en marcha un servidor propio podría ser una solución al aumentar el control sobre la disponibilidad de acceso de los estudiantes.

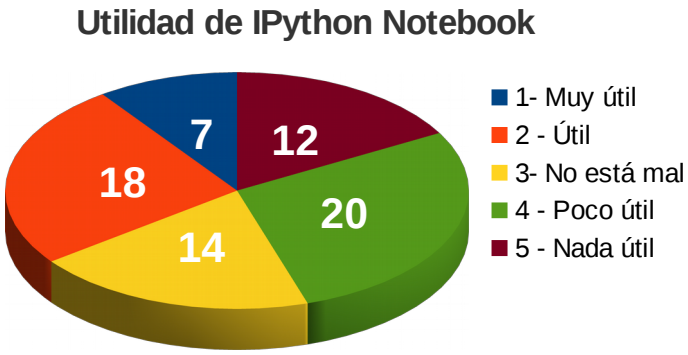
6. Anexos

Anexo I

Se adjuntan los resultados de la encuesta realizada. El número de estudiantes que han contestado a la encuesta ha sido 71.







Anexo II

Se adjuntan los documentos en formato PDF elaborados.

Trabajo Previo Práctica 1

Eduardo Cabrera Granado

1 Introducción a Python (I)

Editar esta celda (haciendo doble click con el ratón) para introducir el nombre y apellidos.

Nombre y Apellidos:

1.1 Contexto

Python se ha convertido en uno de los lenguajes más populares y por tanto usados en muy diferentes áreas. También ha ganado cada vez más terreno en el ámbito científico, gracias a una gran cantidad de módulos científicos de calidad gratuitos y a una sintaxis limpia y fácil de aprender.

En este notebook vamos a ver varios conceptos que nos servirán de base para representar datos, ajustarlos a un módelo (ya sea un modelo lineal o una función más complicada) y a efectuar un análisis de los resultados. Será por tanto una muy breve introducción, fijándonos únicamente en lo que nos sea útil para poder trabajar con Python. Para una explicación más detallada, es muy recomendable consultar alguno de los siguientes enlaces:

- Tutorial de Python
- Dive into Python
- Tutorial de Python en español, con libro en pdf
- Tutorial sobre las herramientas científicas en Python
- Diversas lecciones sobre computación científica con Python

Antes de empezar, una nota para los usuarios de Octave/MATLAB. La sintaxis de las distintas órdenes que usemos son muy similares a las usadas en estos programas, y básicamente la mayor parte de los comandos válidos en aquellos programas son válidos para Python. Para facilitar la transición a Python, se puede consultar el siguiente enlace,

Equivalencia de comandos para usuarios de MATLAB

1.2 IPython Notebook, ¿cómo se utiliza?

Durante el curso utilizaremos en las prácticas el programa IPython, más concretamente su interfaz web IPython Notebook, es decir, se sirve del navegador web para visualizar los resultados. IPython Notebook permite combinar texto con fórmulas matemáticas, código, imágenes e incluso vídeos en un mismo documento por lo que resulta muy útil para trabajar en un proyecto científico o para documentar código. Para poder usarlo desde casa sin necesidad de instalarlo, disponemos de un servicio online que permite editar, ejecutar, salvar y gestionar una biblioteca de notebooks en la red. Este servicio que se llama SageMathCloud es propocionado por la Universidad de Washington y permite tener un espacio personal en donde correr

programas de distintos tipos, incluso editar un documento en LaTeX. También, por supuesto, permite crear, subir, editar y ejecutar notebooks.

Para crear una cuenta en este servicio, se pueden seguir las instrucciones detalladas en la página del Campus Virtual del Laboratorio de Óptica Biomédica.

¿Cómo se utiliza?

El documento se divide en **celdas** que pueden ser de **texto** o de **código** de programación (estas últimas se etiquetan con el símbolo **In [ ]:** a su izquierda. Cuando se ejecutan, aparece un número entre los corchetes que identifica a esa celda).

¿Cómo modifico una celda de texto?

Para modificar un texto se ha de hacer doble-click con el ratón sobre él. Una vez terminada la modificación, y para salir del modo de escritura, se ha de pinchar con el ratón fuera de esa celda o bien *ejecutarla* (al no haber código, la ejecución solo permite mostrar el texto de una forma más legible). Para ejecutar una celda, veáse un poco más abajo.

¿Cómo modifico una celda de código?

Para modificar el código y ver el resultado, se procede de la misma forma: doble-click con el ratón sobre el código, se procede a modificar los valores de los parámetros y posteriormente se ejecuta la celda. A continuación se describe cómo se realiza dicha ejecución.

¿Cómo ejecuto una celda?

Para ejecutar una celda y por ejemplo, ver el resultado del código que en ella se ha escrito, se ha de presionar a la vez *Mayúsculas + Enter* o bien, pinchar con el ratón en el triángulo (símbolo de *Play*) que se encuentra en la barra superior (justo debajo de la opción en el menú indicada por la palabra *Cell*).

¿Cómo creo una celda nueva?

Para generar una nueva celda se puede hacer con el menú superior, pinchando en *Insert* y eligiendo insertar una celda encima o debajo de la celda actual o bien con el comando *Ctrl-m b* para insertar una celda

¿Cómo salvo mi documento?

El documento se autosalva automáticamente cada 2 minutos. Aun así, se puede pinchar en **File** en la barra superior y después en **Save and Checkpoint**. Esto obligará a salvar el documento pero además creará una imagen del documento a la que se puede volver más adelante, por si se hace un cambio que no se desee y se haya autosalvado. Por último, para cerrar el documento limpiamente se ha de pinchar de nuevo en **File** y después en **Close and halt**.

1.3 Uso de Python

Vamos a ver brevemente algunos aspectos de Python que usaremos en esta práctica así como en el resto de actividades que realizaremos con este tipo de documentos. Estas líneas no pretenden ser una guía de Python, únicamente servir para familiarizarse con el Notebook a partir de ejemplos sencillos.

Primero Python puede usarse como una calculadora. Ejecuta las siguientes celdas

```
In []: 5.0*25
```

```
In []: 4.0/7
```

Es necesario indicar que Python distingue entre la división `4.0/7` y `4/7`. La última la toma como una división entre enteros y su resultado será cero (otro entero), mientras que la primera la toma como una división entre números reales. Puedes crear una celda de código a continuación y comprobarlo escribiendo `4/7` y viendo el resultado

```
In []: 3**2
```

Como vemos, Python indica la operación `elevado a` con un doble asterisco. Otros programas como Octave/MATLAB lo hacen con el símbolo `^`.

Variables

Cuando trabajamos con datos, normalmente los usamos más de una vez. Para poder recurrir a ellos cuando queramos, los almacenamos en variables. Por ejemplo,

```
In [3]: long_de_onda = 25
```

Ahora, una vez ejecutada la celda anterior, podemos recurrir a la variable `long_de_onda` cuando queramos.

```
In []: long_de_onda*5
```

Podemos escribir los nombres que queramos (o casi, hay algunos nombres reservados, pero son pocos) para las variables, pero es mejor dar nombres representativos. Si no hemos definido una variable y la llamamos, Python nos dará error.

```
In []: frecuencia
```

Funciones

Una función permite devolver un valor cuando le suministramos un argumento. Por ejemplo, la función `x`^{2} devuelve el cuadrado del número que introduzcamos como argumento `x`. Python viene con múltiples funciones por defecto, pero en ocasiones, querremos definir una función nosotros mismos. Por ejemplo, a la hora de hacer un ajuste no lineal. Veamos cómo hacerlo,

```
In []: def mifuncion(x):
           u = 3*x**2
           v = 5*x
           return u + v
```

En la anterior celda hemos definido una función que hemos llamado `mifuncion` con un único argumento `x` y que devuelve la expresión $3x^2 + 5x$. Hemos usado varias cosas a la vez. Veámoslas una a una:

- Para definir una función comenzamos con la palabra `def` y después escribimos el nombre de la función, poniendo entre paréntesis los argumentos de la misma. Finalmente tenemos que terminar esta primera línea con dos puntos `:`
- A continuación escribimos el cuerpo de la función, es decir, las operaciones que queremos que se realicen cuando llamamos a nuestra función. Esta parte tiene que escribirse con un sangrado para indicar qué va dentro de la función. si a continuación queremos escribir una línea que no sea parte de la función, la escribiremos sin ese sangrado. Python sabrá entonces que eso no pertenece a la función.
- Finalmente, y comenzando con la palabra `return` escribimos lo que queremos que la función devuelva.

Este ejemplo también se podría haber escrito de forma más compacta como,

```
In []: def mifuncion(x):
           return 3*x**2 + 5*x
```

1.4 Ejercicio

- Define una función que tenga como argumentos 3 variables x, y y z y devuelva $x^2 - 3y + z$

1.5 Módulos

Python es un lenguaje de programación de uso general, es decir, no está orientado en un principio a uso científico. Sin embargo, tiene una gran cantidad de módulos que le dan una gran versatilidad y en concreto, muchos que permiten un uso científico de alto nivel. Un módulo no es más que un archivo en donde se definen un conjunto de funciones para realizar distintas operaciones. En otros lenguajes se denominan librerías de funciones. Para hacer uso de esas funciones, tenemos que importar el módulo. Veremos cómo utilizarlos e importarlos más adelante. Por ahora, hay que destacar entre todos 3 esenciales a la hora de realizar cálculos científicos en Python.

- Numpy
- Scipy
- Matplotlib

El primero de ellos, Numpy, proporciona las operaciones básicas para tratar con vectores y matrices, muchas de las funciones básicas como seno, coseno, exponencial, logaritmo, etc., así como algunas funciones para realizar la transformada de Fourier o álgebra lineal por ejemplo. Scipy es un conjunto de submódulos cada uno de los cuales se dedica a un tema distinto de cálculo numérico. Por ejemplo, encontramos en él un submódulo dedicado al procesado de imágenes (es decir, una librería de funciones especializadas en procesado de imágenes), otro dedicado a la optimización, otro distinto dedicado a estadística. Cada uno de estos submódulos contienen una gran cantidad de funciones útiles en estos campos. Para alguien familiarizado con MATLAB, podríamos decir que Scipy es como las toolboxes de este programa comercial, que extienden la funcionalidad del programa en distintos ámbitos.

Finalmente, el módulo Matplotlib es el módulo más utilizado para obtener figuras en Python y es el que utilizaremos para dibujar nuestros resultados.

¿Cómo importar un módulo?

Para importar un módulo se utiliza el comando `import`. Se pueden importar todas las funciones de un módulo o bien únicamente una función determinada.

Aunque es más recomendable importar los módulos uno a uno, existe una forma de indicar al programa que cargue los módulos más utilizados, es decir, Numpy y Matplotlib. El módulo Scipy es algo más avanzado y las funciones que necesitemos de él las importaremos una a una cuando llegue el momento.

Para ello antes de ejecutar cualquier código debemos escribir en una celda el comando:

```
%pylab inline
```

Este comando hace dos cosas a la vez. Por un lado, `%pylab` indica al programa que cargue todas las funciones de Numpy y Matplotlib. Además, añadiendo `inline` le decimos que muestre las figuras incrustadas en el documento.

Así pues, lo añadiremos a cada notebook al principio del documento, lo ejecutaremos y ya tendremos cargados los módulos para el resto de celdas (solo hay que ejecutarlo una vez). Es necesario indicar que si se cierra el notebook y se vuelve a abrir, este comando habrá de ejecutarse otra vez al principio de nuestra sesión.

Con el único fin de mostrar todas las posibilidades, dejamos a continuación cómo sería la importación de todas las funciones de Numpy y de Matplotlib de forma individual (aunque no lo usaremos en en este curso).

```
In []: from numpy import *
       from matplotlib.pyplot import *
```

Tareas Práctica 1

Eduardo Cabrera Granado

Editar esta celda (haciendo doble click con el ratón) para introducir el nombre y apellidos.

Nombre y Apellidos:

Introducción

Para comenzar a trabajar, debemos cargar los módulos de cálculo científico que dan a Python la funcionalidad que queremos. Como se ha comentado en el trabajo previo, importar estos módulos puede hacerse de manera individual, o bien de manera conjunta con el comando `%pylab inline`. Para facilitar la puesta en marcha, utilizaremos esta última forma

```
In [1]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

1 Trabajo con ficheros de datos

Cuando trabajamos en el laboratorio, y queramos analizar las medidas, vamos a necesitar contínuamente cargar los datos obtenidos y realizar operaciones con ellos, como son ajustes a un modelo teórico, además de representar los resultados para poder mostrarlos.

En este apartado de la práctica vamos a aprender cómo hacer este proceso. Supondremos que hemos realizado una cierta medida con un aparato que nos ha devuelto un fichero de datos, normalmente de varias columnas. Veremos cómo cargar esos datos a nuestro programa de Python y realizar algunas operaciones con ellos. Posteriormente, en la siguiente sección, veremos cómo hacer figuras para mostrar los resultados.

En Python (aunque la sintaxis es muy similar en otros programas), para cargar un fichero de datos usaremos la función `loadtxt` que se encuentra dentro del módulo Numpy. Su funcionamiento es muy simple

```
In []: y = loadtxt('prueba_carga_fichero.dat')
```

Como vemos, primero hemos cargado todas las funciones del módulo Numpy para poder usarlas. Después, hemos cargado los datos que se encuentran en el fichero `prueba_carga_fichero.dat` en la variable `y`.

Esta variable es un *array*, es decir, un conjunto de datos. A partir de ahora nos referiremos a ella con este nombre.

Ya está, hemos cargado los datos de nuestro fichero en el programa. Vamos a ver ahora cómo seleccionamos algunos datos (columnas, filas), y algunas operaciones que podemos hacer con ellos.

¿Cuántas filas y columnas tiene la variable `y`?

Esta información la podemos obtener con la función `shape`,

```
In []: shape(y)
```

El primer número nos indica el número de filas (en este caso 1000), y el segundo nos indica el número de columnas (2).

Veamos cómo seleccionar cada una de las columnas,

```
In []: columna1 = y[:,0]
       columna2 = y[:,1]
```

Nota para usuarios de Octave: En Octave/MATLAB se usan paréntesis en vez de corchetes. Además, la primera columna sería etiquetada con el índice 1 en vez de 0.

Veamos ahora cómo seleccionar cada una de las filas,

```
In []: fila1 = y[0,:]
       fila2 = y[1,:]
```

Vemos que las filas se seleccionan con el primer índice, mientras que las columnas se seleccionan con el segundo.

Si queremos visualizar alguna de las filas o columnas, lo podemos hacer con el comando `print`

```
In []: print fila1
```

Nos devuelve un array de dos elementos (porque hay dos columnas)

¿Y si queremos seleccionar solo las primeras 20 filas?

```
In []: filas20 = y[0:20,:]
       print filas20
```

¿Y si queremos seleccionar las primeras 15 filas de la primera columna?

```
In []: filas15col1 = y[0:15,0]
       print filas15col1
```

Estos ejemplos deberían servir para ver el funcionamiento de la selección de diversos elementos de un array en Python.

Ejercicio 1

- Selecciona de la variable `y` únicamente entre la fila 10 y la fila 35 de la segunda columna.

1.1 Otras operaciones con arrays

Evidentemente no sólo podemos seleccionar subconjuntos de los datos cargados. Veamos cómo hacer operaciones con ellos y extraer algunas características,

Suma, resta, multiplicación, etc

La suma, resta, multiplicación, división, elevar a una potencia, o aplicar una función a un array, se realizan como si fueran números. Por defecto Python realiza la operación elemento a elemento.

```
In []: multiplicacion = columna1*columna2

       division = columna1/columna2

       potencia = columna1**2

       suma_col = columna1 + columna2

In []: nuevoarray = 5*cos(columna1)
```

Media, desviación estándar de datos

```
In []: mediacol1 = mean(columna1) # o bien mean(y[:,0])

       desvestad_col1 = std(columna1)#std=sqrt(sum_i[(xi-xmedio)^2/N])

       print mediacol1

       print desvestad_col1
```

1.2 Salvar datos a un fichero

Cuando hemos terminado de procesar los datos, frecuentemente querremos salvar los resultados en un fichero. Esta operación se realiza en Python con la función `savetxt`. Su funcionamiento es el siguiente:

`savetxt('nombredelfichero', variable)`

Es decir, primero damos el nombre del fichero entre comillas simples (o dobles, da igual) y a continuación, como segundo argumento, el nombre de la variable que queremos salvar. Veamos un ejemplo,

```
In []: savetxt('prueba_salvar_datos.dat',suma_col)
```

Lo que nos salvará el resultado de la suma de las dos columnas calculado anteriormente. Si ahora acudimos a la lista de ficheros de nuestro proyecto (arriba a la izquierda en la página de SageMath hay un enlace para verlos), podremos ver el nuevo fichero creado.

Ejercicio 2

- Carga el fichero problema `'problema_datos.dat'` y realiza las siguientes operaciones
  - Calcula la media y la desviación estandar de los datos almacenados en la segunda columna
  - Suma los elementos de cada columna.
  - Salva en un fichero el resultado de elevar al cubo la segunda columna.

2 Figuras

La presentación de los resultados es una tarea esencial en cualquier trabajo, ya sea de laboratorio o de otro tipo. En esta sección vamos a ver cómo podemos hacer figuras en Python. La sintaxis es completamente similar a la utilizada en otros programas como Octave/MATLAB por lo que el paso a estos programas será muy sencillo si se aprende cómo hacerlo en Python.

En esta parte asumiremos que ya tenemos un array o varios que queremos dibujar en una figura.

Supongamos entonces que tenemos dos arrays, `tiempo` y `posicion` por ejemplo, que pueden provenir de cargar un fichero de datos, quizás del fichero utilizado anteriormente `prueba_carga_fichero.dat`.

```
In []: y = loadtxt('prueba_carga_fichero.dat')
       tiempo = y[:,0] # primera columna
       posicion = y[:,1] #segunda columna
```

Ahora queremos visualizar `posicion` frente a `tiempo`. Utilizaremos la función `plot` del módulo Matplotlib, que hemos cargado anteriormente.

```
In []: plot(tiempo,posicion)
```

Vemos que ha salido una función sinusoidal.

Ahora vamos a describir algunas formas de modificar o enriquecer esta figura.

Adición de etiquetas en los ejes

Podemos añadir etiquetas en los ejes mediante los comandos `xlabel` e `ylabel`.

```
In []: plot(tiempo,posicion)
       xlabel('Tiempo (s)')
       ylabel('Posicion (cm)')
```

Cambio de las propiedades de la línea dibujada

Hay varias opciones para cambiar las líneas mostradas. Algunas de ellas son:

- Cambio de anchura: Se añade a plot(tiempo,posición) `linewidth = número` donde número por defecto es 1, así que para aumentar el grosor de la línea habrá que indicar un número mayor (por ejemplo, 2)
- Cambio de color: Se añade `color = nombre del cólor en inglés` por ejemplo 'orange', 'red', 'green', etc.
- Cambio de formato: Si queremos que en vez de una línea contínua sea una línea discontínua añadiremos `linestyle='dashed'`, etc.

```
In []: plot(tiempo,posicion,color='orange')
```

```
In []: plot(tiempo,posicion,color='red',linewidth=2,linestyle='dashed')
```

Nota: Hay que decir que además de la forma explicada para cambiar la línea dibujada, se utiliza mucho una forma compacta para escribir menos. La anterior figura se podría generar con,

```
plot(tiempo,posicion,'r--',lw=2)
```

Cambio de la región representada

¿Qué ocurre si no queremos representar todo el rango que recorren los datos?. Por ejemplo, ¿cómo hacemos para representar únicamente la región de tiempos entre 40 s y 60 s en el ejemplo anterior?.

Para ello debemos modificar el rango de representación de los ejes. Se hace de la siguiente forma,

```
In []: plot(tiempo,posicion,color='red',linewidth=2,linestyle='dashed')
       xlim(40,60)

In []: # Otro ejemplo
       plot(tiempo,posicion,color='red',linewidth=2,linestyle='dashed')
       xlim(40,60)
       ylim(-5,5)
```

Dibujar varias líneas a la vez en una misma figura

Para ello se añaden dentro de plot los distintos pares de vectores x e y a representar, o bien se escribe otra sentencia con plot en la misma celda. Por ejemplo, calculemos la posición al cuadrado y representémosla junto a la posición.

```
In []: pos2 = posicion**2
       plot(tiempo,posicion,tiempo,pos2)

In []: plot(tiempo,posicion,color='red',linewidth=2,linestyle='dashed')
       plot(tiempo,pos2,color='blue',linewidth=2)
```

Subfiguras

Para hacer varias subfiguras podemos utilizar el comando `subplot` varias veces de la siguiente forma.

```
In []: subplot(1,2,1) # subplot(numero filas, numero columnas, figura en la que dibujaremos)
       plot(tiempo,posicion, color='red', linewidth=2)

       subplot(1,2,2) # ahora cambiando el último argumento elegimos la segunda figura para dibujar
       plot(tiempo,posicion-std(posicion), color='black', linewidth=2) # podemos hacer operaciones dentro de plot
```

Si la figura no queda bien de tamaño, podemos modificarla con el comando,

`figure(figsize=(tamañoX, tamañoY))`

Los parámetros de tamaño dependen de la resolución de la pantalla, así que lo mejor es probar con algunos hasta ver la figura de un tamaño adecuado.

```
In []: figure(figsize=(15,4))
       subplot(1,2,1) # subplot(numero filas, numero columnas, figura en la que dibujaremos)
       plot(tiempo,posicion, color='red', linewidth=2)

       subplot(1,2,2) # ahora cambiando el último argumento elegimos la segunda figura para dibujar
       plot(tiempo,posicion-std(posicion), color='black', linewidth=2) # podemos hacer operaciones dentro de plot
```

Mapas de color

Normalmente dibujar unos datos frente a otros no va a ser suficiente a la hora de presentar nuestros resultados. Algunas veces nos encontramos con funciones que dependen de dos variables y que por tanto si las representamos obtendríamos una superficie.

Una forma muy útil de representar este tipo de funciones o datos es mediante los mapas de color, donde a cada valor de la función se le asocia un color. Quizás el ejemplo más cercano pueda ser un mapa de relieve de una zona. En este caso, cada color sería una altura diferente. Combinar este tipo de representación con líneas que marquen las zonas de igual altura también es algo usual y favorece la representación.

En el presente curso, usaremos los mapas de color ampliamente para visualizar las aberraciones en el plano de la pupila de salida del sistema óptico. Al ser un plano, cada punto vendrá determinado por dos variables (x, y), para el cual tendremos un valor de la aberración $W(x, y)$. Este tema será cubierto al final del curso.

En Python, usaremos la función `pcolormesh` para generar mapas de color

```
In []: x,y = mgrid[-10:10:100j,-10:10:100j] # esta instrucción genera las coordenadas (X,Y) del plano
       z = exp(-((x/2.0)**2 + (y/2.0)**2))
       pcolormesh(x,y,z)
```

Trabajo con imágenes

Cuando trabajamos con cámaras CCD necesitaremos cargar en el programa una imagen ya capturada para realizar un análisis de ella, por ejemplo para realizar filtros, ver el tamaño de las estructuras observadas, etc.

Para cargar imágenes se utiliza una función distinta a la de carga de ficheros de datos. Esta función se llama `imread`. Con esta función cargaremos la imagen en una variable, que será un array bidimensional (una matriz). Posteriormente, para visualizar la imagen debemos utilizar la función `imshow`. Veamos cómo funcionan con un ejemplo,

```
In []: particulas = imread('nanopart1.jpg')
       imshow(particulas)
```

Ejercicio 3

- Carga el fichero 'problema_figuras.dat' . Este fichero tiene 3 columnas. La primera se corresponde al tiempo, la segunda a la intensidad de la señal de fluorescencia medida para una cierta muestra, y la tercera, la misma fluorescencia pero para otra muestra diferente. Asigna cada columna a una variable distinta
- A continuación, dibuja la segunda columna frente a la primera, utilizando un rango de representación adecuado, de modo que únicamente se vea el decaimiento de la señal desde su máximo. Indica utilizando las etiquetas que el eje X se corresponde con el tiempo y el eje Y se corresponde con la intensidad.
- En una figura nueva, dibuja en dos subfiguras cada una de las señales de fluorescencia con respecto al tiempo. Añade etiquetas, utiliza un color diferente para cada una de las dos curvas y asigna un grosor de línea de 2 a ambas líneas.

3 Ajuste de datos

Introducción previa

Para hacer un ajuste a unos datos vamos a usar el módulo Scipy. Hay múltiples posibilidades en Scipy a la hora de abordar el ajuste de un conjunto de datos a un cierto modelo, sea este lineal (una recta), polinómico o una función arbitraria. Nos centraremos en cómo hacer una regresión lineal básica con Scipy y en cómo hacer un ajuste a un modelo arbitrario. En ambos casos, lo que subyace en los algoritmos utilizados es minimizar la suma de las distancias de nuestra curva ajuste a nuestros datos en cada punto. Es decir, un ajuste por mínimos cuadrados.

3.1 Regresión lineal

Supongamos que tenemos un conjunto de pares de datos (X, Y), que sabemos (o sospechamos) que siguen una relación lineal. Es decir,

$$Y = mX + b$$

¿Cómo podemos con Python obtener m y b?. Vamos a usar la función `linregress` del submódulo de Scipy dedicado a estadística *Scipy.stats*. Así pues lo primero que debemos hacer es importar esta función. Supondremos que ya tenemos cargados el resto de módulos por haber ejecutado anteriormente el comando `%pylab inline`. Si no es así, incluir este comando en la celda siguiente y ejecutarla.

```
In []: from scipy.stats import linregress
```

La función `linregress` toma como argumentos dos arrays `x` e `y`, que toma como dos conjuntos de medidas que deben tener la misma longitud. Por otro lado, devuelve,

- `slope`, es decir, la pendiente que hemos llamado m anteriormente,
- `intercept`, que es la ordenada en el origen b,
- `r_value` que es el coeficiente de correlación para estimar la bondad de nuestro ajuste.
- `p_value` el cual es un parámetro que nos da la probabilidad de que nuestro resultado se pudiera obtener con un modelo distinto. En este caso, que la pendiente sea nula. Si es menor que un 5% aproximadamente, es que nuestra hipótesis inicial es correcta.
- `stderr` nos da una medida de cuánto se aleja nuestra curva de los puntos experimentales

De la salida de la función, nos interesará los 3 primeros valores, especialmente. Vamos a aplicarlo a un ejemplo. Primero cargaremos los datos de un fichero que tiene dos columnas y asignaremos cada columna a una variable `x` e `y`.

```
In []: prueba = loadtxt('prueba_reglin.dat')
       x = prueba[:,0]
       y = prueba[:,1]
```

Vamos a dibujar nuestros datos antes de realizar el ajuste lineal.

```
In []: plot(x,y,'o')
```

Ahora realizamos el ajuste siguiendo la ayuda de la función `linregress` que nos indica cómo introducir los datos

```
In []: pendiente, ordenada_origen,r,p,stderr = linregress(x,y)
       print 'pendiente = ', pendiente
       print 'ordenada en el origen = ' , ordenada_origen
       print 'coeficiente de correlación r = ', r
```

Ahora vamos a dibujar superpuestos nuestro modelo y nuestros datos experimentales

```
In []: y_modelo = pendiente*x + ordenada_origen

       plot(x,y,'o',x,y_modelo,'r')
       legend(('Datos','Modelo'),loc=0)
```

Uno de los problemas que podemos tener al usar la función `linregress` es obtener el error en la pendiente y la ordenada en el origen. Estos errores no son devueltos por esta función, por lo que deberíamos calcularlos a mano. Sin embargo, también podemos usar otras funciones más completas en Python para obtenerlos. Una de ellas es la función `curve_fit` la cual veremos a continuación.

Ejercicio 4

Carga los datos (2 columnas) del fichero `ejerc_reglin.dat` en una variable, y realiza un ajuste lineal a dichos datos. Muestra el valor de la pendiente y de la ordenada en el origen y justifica si consideras que el ajuste da la tendencia de los datos.

3.2 Ajuste a un modelo arbitrario

Aunque en ocasiones podemos ajustar nuestras medidas para obtener información de un ajuste lineal, en la mayoría de los casos nuestro modelo será algo distinto a una recta. ¿Podemos obtener los parámetros de un modelo con una dependencia arbitraria mediante un ajuste a nuestras medidas experimentales?. La respuesta es, por supuesto, sí, y como se ha comentado anteriormente, Python proporciona múltiples funciones para hacer este tipo de análisis.

Nosotros vamos a centrarnos en la función `curve_fit`, la cual es parte del submódulo `optimize` de Scipy. Vamos a importarla y ver su funcionamiento con el comando help.

```
In []: from scipy.optimize import curve_fit
       help(curve_fit)
```

Esta función es un poco más compleja que `linregress`, pero es el precio que pagamos por tener una herramienta más flexible. Vamos a describir los distintos argumentos de la función y cómo se emplea.

- Argumentos
  - `f` la cual es la función modelo a la que queremos ajustar nuestras medidas. La función `curve_fit` asume que nuestros datos (`x_data`, `ydata`) siguen esta función, es decir, `ydata = f(x_data, *params) + eps`. Aquí `eps` es un cierto error que el programa intentará minimizar, mientras que `*params` son los parámetros de nuestra función (es decir, otros argumentos de la función que no son nuestras medidas experimentales).

  Por ejemplo, si nuestros datos siguen una función gaussiana, $f(x) = ae^{-b(x-x_0)^2}$, los parámetros que variaremos para encontrar aquella que se ajuste a nuestros datos serán a y b.

  - `xdata`,`ydata` son nuestros datos, los que queremos ajustar.
  - `p0` son los parámetros iniciales a partir de los cuales `curve_fit` intenta ajustar la función. Este argumento se puede no dar, en cuyo caso `curve_fit` comienza con los parámetros igual a 1. En nuestro ejemplo de la función gaussiana, los parámetros son [a,b] (escritos como una lista). Si sabemos que, por ejemplo, el valor de a se encuentra próximo a 0.6 y b próximo a 5.8, daríamos como sugerencia al programa `p0 = [0.6,5.8]`. **Dar una sugerencia inicial correcta al programa de ajuste permite que el ajuste se realice más rápidamente. Incluso podría ser que sin ella la función `curve_fit` no sea capaz de encontrar los parámetros adecuados**.
  - `sigma` es un vector que nos da el error de cada una de las medidas de `ydata`. Si no se especifica, se considera que los errores son nulos.
- Salida de `curve_fit`
  - `popt` Como salida de la función `curve_fit` obtenemos primero una lista de los parámetros óptimos que se han encontrado. En nuestro caso, si nuestros parámetros son [a,b], obtendríamos los parámetros óptimos del ajuste [a_{opt}, b_{opt}].
  - `pcov` es la matriz de covarianza de los parámetros. La raíz cuadrada de su diagonal nos proporciona el error (desviación estándar) de cada uno de los parámetros.

Veamos un ejemplo de su uso. Vamos a intentar ajustar la caída de la señal de fluorescencia medida en una muestra de tejido con un tumor maligno a un modelo con una doble exponencial (este modelo se verá en las clases de teoría).

```
In []: y = loadtxt('TumorMalignoCaida.dat')
       tiempo = y[:,0]
       fluorescencia = y[:,1]
       plot(tiempo,fluorescencia,'o')
```

Esta caída sabemos por nuestro modelo teórico que debería seguir una suma de dos exponenciales. Es decir, la función a la que debería ajustarse es,

$$f(t) = c_1 e^{-t/t_1} + c_2 e^{-t/t_2}$$

Aquí los parámetros del modelo serán c_1, c_2, t_1 y t_2. Escritos en una lista serán $[c_1, t_1, c_2, t_2]$. Vamos a definir por tanto nuestra función modelo según esta expresión,

```
In []: def fun_modelo(t,c1,t1,c2,t2): #c1,t1,c2,t2 son los parámetros de nuestro modelo.
           return c1*exp(-t/t1) + c2*exp(-t/t2)
```

Ahora vamos a dar una sugerencia inicial a nuestros parámetros para que a `curve_fit` le sea más sencillo ajustar los datos experimentales

```
In []: c1_ini = 20.0
       c2_ini = 20.0
       t1_ini = 100.0
       t2_ini = 500.0
       params_ini = [c1_ini,t1_ini,c2_ini,t2_ini] # ordenados según los requiere la función modelo.
```

Ya solo nos falta llamar a la función `curve_fit` para que realice el ajuste

```
In []: params_opt, pcov = curve_fit(fun_modelo,tiempo,fluorescencia,p0 = params_ini)
       print params_opt
```

Como vemos, `curve_fit` nos da los parámetros óptimos, ordenados según se los hemos dado y requiere la función modelo. Así,

```
In []: c1_opt = params_opt[0]
       t1_opt = params_opt[1]
       c2_opt = params_opt[2]
       t2_opt = params_opt[3]
```

Vamos a ver ahora el resultado de nuestro ajuste, dibujando en una gráfica los puntos experimentales y nuestro modelo.

```
In []: figure(figsize=(7,4))
       plot(tiempo,fluorescencia,'o',tiempo,fun_modelo(tiempo,c1_opt,t1_opt,c2_opt,t2_opt),'r')
       xlabel('Tiempo (ps)',fontsize=14) # el parámetro fontsize permite modificar el tamaño de la letra utilizada
       ylabel('Fluorescencia ',fontsize=14)
```

Como vemos se ajusta perfectamente. Pero, ¿y si queremos saber los errores asociados a nuestros parámetros?. Esta información nos la da `pcov`, que también es devuelta por `curve_fit` y no hemos usado hasta ahora. El error de nuestros parámetros viene dado por la raíz cuadrada de los elementos de la diagonal de esta matriz.

```
In []: print pcov # Mostramos la matriz pcov
```

```
In []: c1_opt_error = sqrt(pcov[0,0])
       t1_opt_error = sqrt(pcov[1,1])
       c2_opt_error =sqrt(pcov[2,2])
       t2_opt_error = sqrt(pcov[3,3])

       print "Parámetros con su error"
       print "c1 = ", c1_opt , 'Error c1 = ', c1_opt_error
       print "t1 = ", t1_opt , 'ps', 'Error t1 = ', t1_opt_error, 'ps'
       print "c2 = ", c2_opt , 'Error c2 = ', c2_opt_error
       print "t2 = ", t2_opt , 'ps', 'Error t2 = ', t2_opt_error, 'ps'
```

Ejercicio 5

Carga los datos del fichero `ejerc_curvefit.dat`. Separa los datos de cada una de las dos columnas del fichero en dos variables `x` e `y`. Estos datos se deberían ajustar al siguiente modelo,

$$y = ae^{-\left(\frac{x-b}{2c^2}\right)^2} + d$$

(Nota: esta es la función modelo a usar).

1. Representar estos datos en una figura.
2. Escribir la función modelo, tomando como parámetros: a,b,c,d
3. Escribir una sugerencia inicial para dichos parámetros.
4. Llamar a la función `curve_fit` para realizar un ajuste de los datos cargados al modelo generado.
5. Representar en una figura los datos junto al resultado del modelo, y mostrar los valores de los parámetros óptimos junto a su error.

Tareas Práctica 2

Oscar Gómez Calderón, Sonia Melle Hernández

1 Óptica Biomédica. Práctica 2

Modificar esta celda incluyendo la información solicitada

NOMBRE Y APELLIDOS:

FECHA DE ENTREGA:

Ejecutar la siguiente celda para cargar el módulo de cálculo científico (`%pylab inline`).

In []:
```
%pylab inline
```

1.1 Tarea 1. Caracterización de espectros de sustancias biológicas

T1.1. Representación gráfica de los espectros de la muestra de colorante elegida

T1.1.a) Transmitancia en función de la longitud de onda entre 400 y 700 nm. Para representar la transmitancia de la muestra de colorante elegida, incluye en la siguiente celda de código el nombre de tu fichero de datos. El fichero de datos habrá sido importado a la nube previamente en el directorio del alumno donde esté este documento.

In []:
```
# MODIFICAR CELDA Y EJECUTAR
filenamecolorante = 'azuldemetileno.txt'
```

In []:
```
# NO MODIFICAR ESTA CELDA, SOLO EJECUTAR

import StringIO
s = open(filenamecolorante).read().replace(',','.'); #Cambia las comas de fichero de datos por puntos
data = loadtxt(StringIO.StringIO(s))
```

Ejercicio 1

El fichero de datos importado tiene dos columnas. En la variable `data` hemos almacenado estas dos columnas. En la primera, está la información de la longitud de onda en (nm). En la segunda columna, tenemos la transmitancia de la muestra de colorante medida en %. Para obtener la transmitancia de la muestra de colorante entre 0 y 1, hay que dividir la segunda columna por 100. Para contestar a este ejercicio añade una celda de código para:

- Asignar cada una de estas columnas a variables (con nombres adecuados). Recordar que para obtener la transmitancia a partir de los datos de la segunda columna, hay que dividir ésta por 100.

- Representar la transmitancia frente a la longitud de onda en una figura. Escribe etiquetas para cada eje y por último, fija el rango de representación del eje X desde 400 nm a 700 nm y el del eje Y desde 0 a 1.

Ejercicio 2

- Escribir en esta celda la expresión que relaciona la absorbancia $A(\lambda)$ con la transmitancia $T(\lambda)$

$$A(\lambda) = ...T(\lambda)$$

- Crear una celda nueva a continuación para:
  - Definir una nueva variable `absorbancia` calculada como el logaritmo en base 10 de la transmitancia. Notas: la función que calcula el logaritmo en base 10 de un número `x` se escribe `log10(x)`. Recordar además que la transmitancia es un valor que va entre 0 y 1.
  - Dibujar la absorbancia frente a la longitud de onda. Escribir etiquetas para cada eje y fijar el rango de representación del eje X desde 400 nm a 700 nm.

T1.2. Relacionar ambas magnitudes: transmitancia - absorbancia

Ejercicio 3

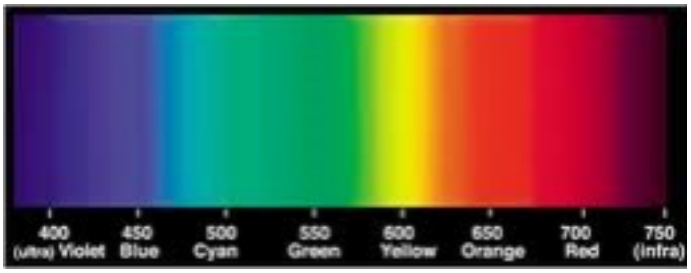
- Relacionar las regiones espectrales de máxima y mínima transmitancia con las de máxima y mínima absorbancia. Escribir en esta misma celda la respuesta.

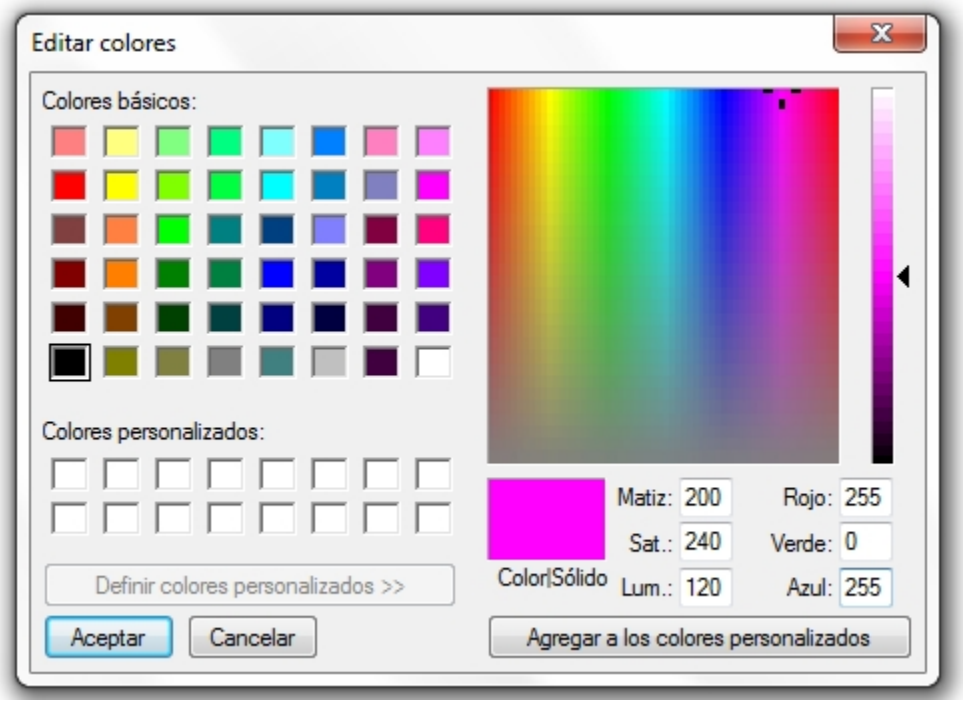
T1.3. Relacionar cualitativamente el espectro con el color aparente del colorante seleccionado

Ejercicio 4

Comentar el rango de longitudes de onda para el que la transmitancia del colorante seleccionado es máxima y la relación de dicho rango con el color de la sustancia. Justifique cualitativamente su respuesta.

Para justificarlo puede, si desea, utilizar el modelo de representación del color en RGB [R G B] (de las siglas en inglés Red, Green, Blue) que asocia a cada color un valor de los tres colores primarios rojo, verde y azul definido entre 0 y 255. En este modelo, el color rojo tomará valores [255 0 0]; el verde [0 255 0]; y el azul [0 0 255]. Se puede utilizar un programa de representación gráfica (como el PAINT) para obtener de forma cualitativa el color aparente de la solución teniendo en cuenta el rango espectral en el que transmite. Por ejemplo, si una sustancia transmite por igual en las longitudes de onda del rojo y el azul, su color será el obtenido por suma de los dos colores primarios rojo y azul, es decir, fucsia [255 0 255], tal como se muestra en el ejemplo siguiente.





Escribir en esta celda la respuesta al ejercicio 4. Si se desea incluir la imagen del editor de colores mostrando el resultado (como aparece en la figura superior), importar en una nueva celda la imagen que se desea incluir (con la función `imread('nombre_archivo.extension')`) y mostrarla con el comando `imshow`. Esta imagen habrá sido subida al espacio de SageMath Cloud previamente en el directorio del alumno.

T1.4. Representación de la absorbancia de la muestra de clorofila

Ejercicio 5

Representamos ahora la dependencia con la longitud de onda de la absorbancia de su muestra de clorofila disuelta en alcohol en el rango de longitudes de onda del visible entre 400 y 700 nm.

En la siguiente celda de código escribir el nombre del fichero de datos correspondiente a la clorofila. El fichero de datos habrá sido importado a la nube previamente en el directorio del alumno donde esté este documento.

In []:
```
# MODIFICAR CELDA Y EJECUTAR
filenameclor = 'perejil.txt'
```

In []:
```
# NO MODIFICAR ESTA CELDA, SOLO EJECUTAR

import StringIO
s = open(filenameclor).read().replace(',','.')
data = loadtxt(StringIO.StringIO(s))
lambdaclor = data[:,0]; transmittanceclor = data[:,1]/100
absorbanceclor=-log10(transmittanceclor)
plot(lambdaclor,absorbanceclor,'ok')
xlabel('$\lambda$ (nm)')
ylabel('Absorbancia')
xlim(400,700)
title('Experimental')
```

In []:
```
#SE PUEDE MODIFICAR ESTA CELDA PARA INCLUIR OTRO FICHERO ENCONTRADO EN LA LITERATURA

img=imread('clorofila.jpg')
imgplot=imshow(img);title('Literatura')
```

T1.5. Interpretación del espectro medido

Ejercicio 6

Compare el espectro medido con el encontrado en la literatura (mostrado mas arriba) decidiendo qué tipo de clorofila (A o B) se encuentra en la sustancia analizada. Modificar esta celda para responder a este ejercicio.

1.2 Tarea 2. Comprobación experimental de la ley de Lambert-Beer

T2.1. Representación de la transmitancia del permanganato potásico de todas las disoluciones

Ejercicio 7

Representar en la misma gráfica la dependencia con la longitud de onda de la transmitancia para cada una de las disoluciones de permanganato potásico medidas en el laboratorio (incluida la muestra problema) en el rango de longitudes de onda del visible (entre 435 y 700 nm).

En la siguiente celda de código escribir el nombre de los fichero de datos correspondientes a las distintas disoluciones. Tener en cuenta que el primer fichero corresponde a la concentración $C = 1 \times 10^{-4}$ mol/l, el segundo fichero a la concentración $C = 2 \times 10^{-4}$ mol/l, ..., el quinto fichero a la concentración $C = 5 \times 10^{-4}$ mol/l, y el útlimo fichero a la concentración problema. Los ficheros de datos habrán sido importados a la nube previamente en el directorio del alumno.

In []:
```
# MODIFICAR CELDA Y EJECUTAR
filenameC1 = 'C1.txt' # C=1 x 10^-4 mol/l
filenameC2 = 'C2.txt' # C=2 x 10^-4 mol/l
filenameC3 = 'C3.txt' # C=3 x 10^-4 mol/l
filenameC4 = 'C4.txt' # C=4 x 10^-4 mol/l
filenameC5 = 'C5.txt' # C=5 x 10^-4 mol/l
filenameCproblema = 'Cproblema.txt'
```

In []:
```
# NO MODIFICAR ESTA CELDA, SOLO EJECUTAR
import StringIO
import mpld3     # el módulo `mpld3` permite mostrar las figuras con la opción de hacer zoom en ellas
from mpld3 import plugins
mpld3.enable_notebook()

s = open(filenameC1).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
lambdaC1 = data[:,0]; transmittanceC1 = data[:,1]/100
s = open(filenameC2).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
lambdaC2 = data[:,0]; transmittanceC2 = data[:,1]/100
s = open(filenameC3).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
lambdaC3 = data[:,0]; transmittanceC3 = data[:,1]/100
s = open(filenameC4).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
lambdaC4 = data[:,0]; transmittanceC4 = data[:,1]/100
s = open(filenameC5).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
lambdaC5 = data[:,0]; transmittanceC5 = data[:,1]/100
s = open(filenameCproblema).read().replace(',','.'); data = loadtxt(StringIO.StringIO(s))
```

```
lambdaCproblema = data[:,0]
transmittanceCproblema = data[:,1]/100

fig=figure(figsize=(10,5))
plot(lambdaC1,transmittanceC1,'y',label='C=0.0001 mol/l')
plot(lambdaC2,transmittanceC2,'g',label= 'C=0.0002 mol/l')
plot(lambdaC3,transmittanceC3,'b',label= 'C=0.0003 mol/l')
plot(lambdaC4,transmittanceC4,'m',label= 'C=0.0004 mol/l')
plot(lambdaC5,transmittanceC5,'r',label= 'C=0.0005 mol/l')
plot(lambdaCproblema,transmittanceCproblema,'k',label= 'C problema')
legend(loc='upper left',bbox_to_anchor=(0.7,0.6),frameon=False)
xlabel('Longitud de onda (nm)',fontsize=16)
ylabel('Transmitancia',fontsize=16)
xlim(435,700)
plugins.connect(fig, plugins.MousePosition(fontsize=14))
mpld3.display(fig)
```

Ejercicio 8

Observando todos los espectros medidos indicar aproximadamente cual puede ser el valor (o el rango) de la concentración problema. Responder a esta cuestión dando una explicación razonada en esta celda de texto.

$$C_{problema} = ... \times 10^{-4} mol/l$$

Nota: En la gráfica anterior se puede hacer zoom con el ratón. El botón para ello aparece al pasar el ratón por encima de la esquina inferior izquierda de la figura. También en la esquina inferior derecha aparecen las coordenadas en las que se encuentra el ratón al moverlo sobre la gráfica.

Escribir a continuación la letra asignada al vial que contenía su muestra problema en el laboratorio (es necesario aportar este dato):

T2.2. Transmitancia mínima de las distintas disoluciones

Ejercicio 9

Vamos a localizar la transmitancia a una longitud de onda concreta λ_0 para cada una de las concentraciones de permanganato potásico. Para ello nos fijaremos en la región del espectro visible donde la transmitancia es mínima pues en esta zona el error cometido será menor. Seguimos los siguientes pasos:

1. Elegir un fichero de datos correspondiente a una concentración.
2. Localizar el valor mínimo de la transmittancia en el rango del visible (entre 435 y 700 nm) y su correspondiente longitud de onda (λ_0).
3. Para el resto de muestras buscar el valor de la transmitancia a esa misma longitud de onda λ_0.

Para localizar el mínimo de la transmitancia o el valor de la transmitancia a la longitud de onda deseada hacer zoom en la figura anterior (para ello pinchar en el símbolo de la lupa que aparece en la parte izquierda inferior de la figura cuando pasamos el ratón sobre ella).

Alternativamente, podemos abrir los ficheros de datos en un editor de texto (Notepad) y leer directamente los valores que aparecen en dos columnas: 1a columna: longitud de onda, 2a columna: transmitancia. Recordar que la transmitancia en los ficheros aparece en tanto por ciento, así que habrá que dividir dicho valor por 100.

- Escribir modificando este celda el valor de la longitud de onda seleccionada (que corresponde a la mínima transmitancia de una muestra).

$$\lambda_0 = ...$$

Ahora vamos a escribir en la siguiente celda de código los valores de las transmitancias encontrados para cada disolución. Tener en cuenta el orden: $Trans1$ corresponde a la concentración $C = 1 \times 10^{-4}$ mol/l, $Trans2$ corresponde a la concentración $C = 2 \times 10^{-4}$ mol/l, ..., $Trans5$ corresponde a la concentración $C = 5 \times 10^{-4}$ mol/l, y $Transproblema$ corresponde a la disolución problema.

```
In []: # MODIFICAR CELDA Y EJECUTAR
       Trans1=0.568 # C=1 x 10^-4 mol/l
       Trans2=0.308 # C=2 x 10^-4 mol/l
       Trans3=0.177 # C=3 x 10^-4 mol/l
       Trans4=0.0997 # C=4 x 10^-4 mol/l
       Trans5=0.0533 # C=5 x 10^-4 mol/l
       Transproblema=0.0997 # concentración problema
```

T2.2. Absorbancia máxima de las distintas disoluciones. Ajuste lineal

A partir de los datos de la transmitancia mínima obtenida en el apartado anterior (T2.1), el programa calcula la absorbancia máxima asociada a cada concentración, utilizando la expresión del apartado T1.1.b).

```
In []: # NO TOCAR, SOLO EJECUTAR
       concentration = array([1e-4, 2e-4, 3e-4, 4e-4, 5e-4])
       transmittancemin = array([Trans1, Trans2, Trans3, Trans4, Trans5])
       absorbancemax=-log10(transmittancemin)
```

Ejercicio 10

- Representar dicha absorbancia máxima (la variable `absorbancemax`) frente a la concentración C (mol/l) (`concentration`) en una nueva celda que se deberá crear a continuación.

Según la ley de Lambert-Beer la absorbancia es proporcional a la concentración del colorante según $A(\lambda_0) = \epsilon_m(\lambda_0) * L * C$, siendo $\epsilon_m(\lambda_0)$ el coeficiente de extinción molar, y L la longitud de la cubeta que contiene a la disolución (1 cm).

- Comprobar dicha ley realizando un ajuste lineal de los datos utilizando la función `linregress`. Representar en la misma gráfica los datos experimentales junto con el ajuste obtenido e indicar el valor de la pendiente. Realizar este ajuste en la celda a continuación de este enunciado. Se ha incluido en ella el comando que importa la función `linregress`.

```
In []: from scipy.stats import linregress
```

T2.3. Coeficiente de extinción molar del permanganato potásico

Ejercicio 11

Empleando el valor de la pendiente del apartado anterior (T2.2) y la expresión $A(\lambda_0) = \epsilon_m(\lambda_0) * L * C$, podemos obtener el valor del coeficiente de extinción molar del permanganato potásico para la longitud de onda de máxima absorción $\epsilon_m(\lambda_0)$. Escribir en esta celda la expresión analítica utilizada para obtener $\epsilon_m(\lambda_0)$.

$$\epsilon_m(\lambda_0) = ...pendiente.....L.....$$

Escribir el valor numérico obtenido con sus unidades

$$\epsilon_m(\lambda_0) = ...$$

Compare el valor obtenido con el encontrado en la literatura para la misma longitud de onda. Modificar esta celda para responder a la pregunta.

T2.4. Estimación de la concentración de la disolución problema

Ejercicio 12

A partir de lo anterior, vamos a estimar la concentración de la muestra de permanganato potásico de concentración desconocida suministrada en el laboratorio. Ejecutando la siguiente celda de código calculamos la absorbancia máxima (a la longitud de onda λ_0) de la muestra problema.

```
In []: # NO MODIFICAR ESTA CELDA, SOLO EJECUTAR
       print 'Absorbancia máxima para la muestra problema=',-log10(Transproblema)
```

Con dicho valor estimar la concentración. Escribir en esta celda la expresión analítica utilizada para obtener la concentración de la muestra problema.

$$C_{problema} = ...A(\lambda_0)....pendiente....$$

Escribir el valor numérico obtenido con sus unidades

$$C_{problema} = ...$$

Comentar si el valor numérico obtenido se corresponde con la estimación llevada a cabo en el apartado T2.1. Modificar esta celda para contestar esta cuestión.

Trabajo Previo Práctica3

Sonia Melle Hernández, Oscar Gómez Calderón

1 Óptica Biomédica. Trabajo previo Práctica 3

NOMBRE Y APELLIDOS:

FECHA DE ENTREGA:

Modificar esta celda incluyendo la información solicitada

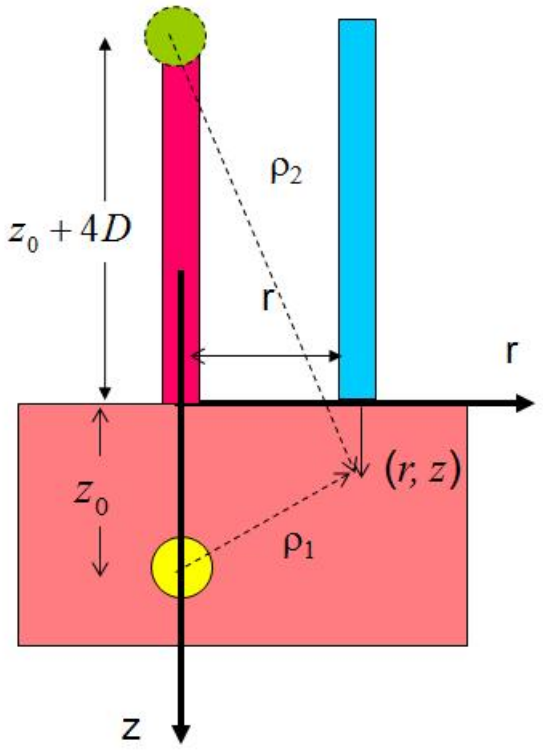
Ejecutar la siguiente celda para cargar el módulo de cálculo científico (`%pylab inline`).

```
In [1]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

Espectro de reflectancia difusa

En la espectroscopía de reflectancia difusa local se ilumina el tejido con una fibra óptica y se recoge la radiación retro-difundida con otra fibra situada a una distancia r de la primera, tal como indica la figura.



Teniendo en cuenta la ecuación de difusión, podemos calcular la reflectancia difusa medida por la fibra de captación como:

$$R_d = Cte\ T_{epi}\ \left[\frac{z_0\,(1+\mu_{eff}\rho_1)}{4\pi\rho_1^3}e^{-\mu_{eff}\rho_1} + \frac{(z_0+4AD)\,(1+\mu_{eff}\rho_2)}{4\pi\rho_2^3}e^{-\mu_{eff}\rho_2}\right]$$

donde Cte es una constante global de normalización. $\mu_{eff} = \sqrt{3\mu_a(\mu_a+\mu_s')}$ es el coeficiente de extinción efectivo de la radiación al propagarse en el tejido, donde μ_s' es el coeficiente de scattering reducido, y μ_a es el coeficiente de absorción (todos en cm^{-1}). $D = 1/(3(\mu_a+\mu_s'))$ es el coeficiente de difusión. $z_0 = 1/(\mu_a+\mu_s')$ es la longitud de penetración de la radiación en el tejido, que es considerada en el modelo de difusión como la distancia en el interior del tejido a la cual se situa la fuente de luz. Sabiendo que la distancia entre las fibras de iluminación y detección es r (en cm), las distancias entre el detector y las dos fuentes, la que se coloca en el interior del tejido (a una distancia z_0 de la frontera) y la imaginaria negativa que se coloca fuera (a una distancia z_0+4D de la superficie) para ajustar la condición de contorno y extender el problema a un medio infinito, son:

$$\rho_1 = \sqrt{z_0^2 + r^2},$$

$$\rho_2 = \sqrt{(z_0+4AD)^2 + r^2}.$$

A está relacionado con el cambio de índice de refracción en la frontera, entre el tejido y el medio externo: $A = (1+R_{eff})/(1-R_{eff})$, con $R_{eff} = -1.440/n_r^2 + 0.71/n_r + 0.668 + 0.0636 n_r$, siendo n_r el índice de refracción del tejido normalizado por el índice del medio externo. T_{epi} es la transmitancia de la epidermis que tiene en cuenta la absorción de la melanina

$$T_{epi} = e^{-Mel\ \mu_{a\ mel}\ L_{epi}}$$

donde Mel es el contenido de melanina en la epidermis, $L_{epi} = 0.006 * 2$ cm es el camino recorrido por la luz en la epidermis al entrar y salir de ella (considerando 60 μm de epidermis), y por último $\mu_{a\ mel}$ es el coeficiente de absorción de la melanina (en cm^{-1}):

$$\mu_{a\ mel}(cm^{-1}) = \frac{6.6 \times 10^{11}}{[\lambda(nm)]^{3.33}}$$

El coeficiente de scattering reducido $\mu'_s = a(\mu'_{s\ Mie} + \mu'_{s\ Ray.})$ es una combinación de scattering Mie y Rayleigh (en cm^{-1}):

$$\mu'_{s\ Mie}(cm^{-1}) = 4.59 \times 10^3\ [\lambda(nm)]^{-0.913}$$
$$\mu'_{s\ Ray.}(cm^{-1}) = 1.74 \times 10^{12}\ [\lambda(nm)]^{-4}$$

mientras que a es una constante que varía la amplitud del scattering.

El coeficiente de absorción (en cm^{-1}) es debido principalmente a la hemoglobina y en menor media al agua ya que mediremos en el visible:

$$\mu_a = B\ (S\ \mu_{a\ oxi} + (1 - S)\ \mu_{a\ desoxi}) + W\ \mu_{a\ water}$$

donde B es la fracción de volumen de la sangre en el tejido, S es el grado de oxigenación de la sangre, $\mu_{a\ oxi}$ y $\mu_{a\ desoxi}$ son los coeficientes de absorción de la oxyhemoglobina y desoxyhemoglobina en la sangre, respectivamente, W es la fracción de volumen de agua en el tejido y $\mu_{a\ water}$ el coeficiente de absorción del agua.

Ejercicio 1

Representar el espectro del coeficiente de absorción de la hemoglobina, la oxihemoglobina y el agua en el rango de 450 a 800 nm. Los ficheros de datos se llaman: agua450-800nm.DAT, hemoglobina450-800nm.DAT y oxihemoglobina450-800nm.DAT y están cargados en este mismo directorio. Estos ficheros tienen dos columnas: en la primera está la longitud de onda en nanómetros, y en la segunda, el coeficiente de absorción en cm-1. Comentar la región espectral donde se producen los máximos de absorción en cada caso.

Ejercicio 2

Representar el espectro del coeficiente de absorción de la melanina en el rango entre 450 y 800 nm. Comentar cuál es la región de máxima absorción.

$$\mu_{a\ mel}(cm^{-1}) = \frac{6.6 \times 10^{11}}{[\lambda(nm)]^{3.33}}$$

Ejercicio 3

Representar el espectro del coeficiente de scattering reducido $\mu'_s = a(\mu'_{s\ Mie} + \mu'_{s\ Ray.})$ considerando a =1 en el rango entre 450 y 800 nm. Representar en la misma gráfica la contribución del scattering Mie y del scattering Rayleigh. Comentar cuál es la región donde se produce mayor scattering y donde domina cada una de las contribuciones (Mie y Rayleigh)

$$\mu'_{s\ Mie}(cm^{-1}) = \frac{4.59 \times 10^3}{[\lambda(nm)]^{0.913}}$$
$$\mu'_{s\ Ray.}(cm^{-1}) = \frac{1.74 \times 10^{12}}{[\lambda(nm)]^4}$$

A continuación vamos a representar la función teórica que calcula la reflectancia difusa R_d dada más arriba. Para ello vamos a utilizar los siguientes valores de los parámetros:

- **r**: distancia entre la fibra que ilumina y la fibra que recoge la luz difundida en el medio. Consideramos 400 micras=0.04cm
- **W**: fracción de volumen de agua en los tejidos. El valor típico del contenido de agua en los tejidos es de 0.65
- **n_r**: índice de refracción del tejido / índice del medio externo. Considerar que el índice del tejido es 1.4.

El resto de parámetros: **B**, **S**, **Mel**, **a**, y **Cte** se utilizarán en la práctica como parámetros de ajuste, es decir, el ajuste de la función teórica a los datos experimentales se realizará variando dichos parámetros hasta conseguir que la curva teórica se aproxime lo máximo posible a la curva experimental. Así obtendremos los valores finales de dichos parámetros. Antes de realizar la práctica vamos a representar la función reflectancia difusa R_d teórica para unos valores de los parámetros razonables.

- **Cte**: constante global de la reflectancia difusa teórica. Vamos a considerarla 0.01.
- **a**: constante que varía la amplitud total del scattering. Vamos a considerar un valor 1.

Ejercicio 5

Leer el artículo "Rapid and accurate estimation of blood saturation, melanin content, and epidermis thickness from spectral diffuse reflectance", Dmitry Yudovsky and Laurent Pilon, Applied Optics vol. 49, pags. 1707-1719. que podéis encontrar en este directorio. Anotar junto a cada uno de los siguientes parámetros cuál es el rango de valores que pueden tomar (ver Tabla 1 del artículo).

- **B**: fracción de volumen de sangre en el tejido. Rango de valores
- **S**: grado de oxigenación. Fracción de sangre oxigenada respecto del total. Rango de valores
- **Mel**: contenido de melanina en la epidermis. Rango de valores

Ejercicio 6

Representar la función reflectancia difusa para valores razonables de los parámetros.

In [30]:
```
# MODIFICAR CELDA Y EJECUTAR
r = 0.04          # cm, separación fibra fuente-fibra detector
W = 0.65          # Contenido en agua del téjido
n_r = 1.4         # índice de refracción del tejido dividido por el del medio externo
Cte = 0.01        # Constante global de la reflectancia difusa teórica
a= 1              # Constante que escala la amplitud total del scattering del tejido.
Lepi =0.006*2     # cm, longitud de la epidermis

B =      # Fracción de volumen en sangre
S =        # Grado de oxigenación
Mel =      # Contenido de melanina en la epidermis


R_eff = - 1.440/n_r**2 + 0.710/n_r + 0.668 + 0.0636*n_r
A = (1 + R_eff)/(1 - R_eff)
Tepi = exp(-Mel*muamel*Lepi)
mua  = B*(S*muaoxy + (1-S)*muadeoxy) + W*muawater
mueff = sqrt(3*mua*(mua+musp))
D=1/(3*(mua+musp))
z0 = 1.0/(mua + musp)
r1 = sqrt(z0**2 + r**2)
r2 = sqrt((z0 + 4*A*D)**2 + r**2)
```

```
RD = Cte*Tepi*(((z0*(1+mueff*r1))/(4*pi*r1**3))*exp(-mueff*r1) +
               ((z0+4*A*D)*(1+mueff*r2)/(4*pi*r2**3))*exp(-mueff*r2))
```

Ejercicio 7

Describir que rasgo característico de este curva podrías asociar con la presencia de sangre oxigenada.

Tareas Práctica 3

Sonia Melle Hernández, Oscar Gómez Calderón

1 Óptica Biomédica. Práctica 3

NOMBRE Y APELLIDOS:

FECHA LIMITE DE ENTREGA:

Modificar esta celda incluyendo la información solicitada

Ejecutar la siguiente celda para cargar el módulo de cálculo científico (`%pylab inline`).

```
In []: %pylab inline
```

1.1 Tarea 1. Determinar la profundidad de penetración de la luz en tejidos biológicos

T1.1. Espectro de reflectancia del cromóforo

Representación gráfica de los espectros de reflectancia. Se representará en la misma gráfica el espectro de reflectancia de un papel tintando (que simula un cromóforo de la piel) junto con el espectro de reflectancia de un papel blanco en el rango de longitudes de onda de 425 a 900 nm.

En la siguiente celda de código escribir el nombre de los ficheros de datos correspondientes al papel tintado y al papel blanco. Dichos ficheros habrán sido importados a la nube previamente en el directorio del alumno.

```
In []: # MODIFICAR SOLO EL NOMBRE DEL FICHERO EN ESTA CELDA Y EJECUTAR
       filenamepapel = 'verde.txt'   #nombre del fichero del papel tintado
       filenameblanco = 'patron.txt'  #nombre del fichero del papel blanco
```

```
In []: # NO TOCAR ESTA CELDA, SOLO EJECUTAR

       import StringIO
       s = open(filenamepapel).read().replace(',','.')
       data = loadtxt(StringIO.StringIO(s))

       lambdapapel = data[:,0]
       reflectancepapel = data[:,1]/100

       s = open(filenameblanco).read().replace(',','.')
       data = loadtxt(StringIO.StringIO(s))

       lambdablanco = data[:,0]
       reflectanceblanco = data[:,1]/100
```

```
fig=figure(figsize=(7,5))
plot(lambdapapel,reflectancepapel,'ro')
plot(lambdablanco,reflectanceblanco,'ko')
legend(('crom','blanco'),loc='lower right')
xlabel('Longitud de onda (nm)')
ylabel('Reflectancia')
xlim(425,900);
```

Ejercicio 1

Análisis de los espectros de reflectancia.

Relacionar la región de longitudes de onda de mayor reflectancia con el color del papel. Escribir en esta celda la respuesta.

T1.2. Espectro de reflectancia del cromóforo cubierto por una lámina de tejido

Representación gráfica de los espectros de reflectancia. Se representarán tres gráficas:

- espectro de reflectancia del papel blanco con y sin la capa de tejido (lámina de ternera) entre 425 y 900nm.
- espectro de reflectancia del cromóforo (papel tintado) con y sin la capa de tejido (lámina de ternera) entre 425 y 900nm.
- espectro de reflectancia del cromóforo y del papel blanco ambos con el tejido superpuesto entre 425 y 900nm.

En la siguiente celda de código escribir el nombre de los ficheros de datos correspondientes al papel tintado y al papel blanco con 1 lámina de ternera. Dichos ficheros habrán sido importado a la nube previamente en el directorio del alumno.

In []:
```
# MODIFICAR SOLO EL NOMBRE DEL FICHERO EN ESTA CELDA Y EJECUTAR
filenamepapelfilete1 = 'verdefilete1.txt'  #nombre del fichero del papel tintado con 1 lámina de ternera
filenameblancofilete1 = 'patronfilete1.txt' #nombre del fichero del papel blanco con 1 lámina de ternera
```

In []:
```
# NO TOCAR, SOLO EJECUTAR

import StringIO
s = open(filenameblancofilete1).read().replace(',','.')
data = loadtxt(StringIO.StringIO(s))
lambdablancofilete1 = data[:,0]
reflectanceblancofilete1 = data[:,1]/100

s = open(filenamepapelfilete1).read().replace(',','.')
data = loadtxt(StringIO.StringIO(s))
lambdapapelfilete1 = data[:,0]
reflectancepapelfilete1 = data[:,1]/100


fig=figure(figsize=(14,7))

subplot(1,3,1)
plot(lambdablanco,reflectanceblanco,'ko',label='blanco')
plot(lambdablancofilete1,reflectanceblancofilete1,'k+',label='blanco+1filet.')
xlabel('Longitud de onda (nm)')
```

```
ylabel('Reflectancia')
title('Papel blanco')
xlim(425,900)
ylim(0,1)
legend(loc='lower right')

subplot(1,3,2)
plot(lambdapapel,reflectancepapel,'ro',label='crom.')
plot(lambdapapelfilete1,reflectancepapelfilete1,'r+',label='crom.+1filet.')
xlabel('Longitud de onda (nm)')
ylabel('Reflectancia')
title('Papel tintado')
xlim(425,900)
ylim(0,)
legend(loc='lower right')

subplot(1,3,3)
plot(lambdapapelfilete1,reflectancepapelfilete1,'r+',label='crom.+1filet.')
plot(lambdablancofilete1,reflectanceblancofilete1,'k+',label='blanco+1filet.')
xlabel('Longitud de onda (nm)');ylabel('Reflectancia');xlim(425,900);ylim(0,);legend(loc='lower right')
```

Ejercicio 2

Análisis del espectro de reflectancia de la muestra patrón.

Explicar cómo se ha modificado el espectro de reflectancia del papel de color blanco cuando se coloca sobre él una lámina de ternera. Discutir en qué rango de longitudes de onda se observan los picos de absorción debidos a la hemoglobina que contiene el filete. Escribir en esta celda la respuesta.

Ejercicio 3

Análisis del espectro de reflectancia del papel tintado.

A) Explicar cómo se ha modificado el espectro de reflectancia del papel tintado cuando se coloca sobre él una lámina de ternera. Escribir en esta celda la respuesta.

B) ¿Hay alguna característica del espectro del papel tintado que se mantenga cuando colocamos sobre él la lámina de ternera? Esta característica nos permitiría determinar si parte de la luz reflejada proviene del cromóforo (papel tintado) oculto tras el tejido. Escribir en esta celda la respuesta.

C) Comparando los espectros correspondientes al papel blanco y al tintado (cromóforo) cuando están cubiertos por la lámina de ternera, explicar si parte de la luz reflejada proviene del cromóforo (papel tintado). Escribir en esta celda la respuesta.

T1.3. Espectro de reflectancia del cromóforo cubierto por dos o mas capas de tejido

Representación gráfica de los espectros de reflectancia. Se representarán las siguiente gráficas:

- espectro de reflectancia del papel blanco sin tejido y con cada una de las capas de tejido interpuestas entre 425 y 900nm.
- espectro de reflectancia del cromóforo (papel tintado) sin tejido y con cada una de las capas de tejido interpuestas entre 425 y 900nm.

- espectro de reflectancia del cromóforo y del papel blanco ambos con dos láminas de tejido entre 425 y 900nm
- espectro de reflectancia del cromóforo y del papel blanco ambos con tres láminas de tejido entre 425 y 900nm
- espectro de reflectancia del cromóforo y del papel blanco ambos con cuatro láminas de tejido entre 425 y 900nm

En la siguiente celda de código escribir el nombre de los ficheros de datos correspondientes al papel tintado y al papel blanco con 2 láminas de ternera. Dichos ficheros habrán sido importado a la nube previamente en el directorio del alumno. Si el alumno ha medido la reflectancia con un número mayor de capas de tejido, debe añadir las celdas de código necesarias para poder representar la reflectancia de todas las capas medidas.

In []:
```
# MODIFICAR CELDA Y EJECUTAR
filenamepapelfilete2 = 'verdefilete2.txt'   #nombre del fichero del papel tintado con 2 láminas de ternera
filenameblancofilete2 = 'patronfilete2.txt'   #nombre del fichero del papel blanco con 2 láminas de ternera
```

In []:
```
# NO TOCAR, SOLO EJECUTAR

import StringIO

s = open(filenameblancofilete2).read().replace(',','.')
data = loadtxt(StringIO.StringIO(s))
lambdablancofilete2 = data[:,0]
reflectanceblancofilete2 = data[:,1]/100

s = open(filenamepapelfilete2).read().replace(',','.')
data = loadtxt(StringIO.StringIO(s))
lambdapapelfilete2 = data[:,0]; reflectancepapelfilete2 = data[:,1]/100

fig=figure(figsize=(14,7))
subplot(2,2,1)
plot(lambdablanco,reflectanceblanco,'ko',label='blanco')
plot(lambdablancofilete1,reflectanceblancofilete1,'k+',label='blanco+1filet.')
plot(lambdablancofilete2,reflectanceblancofilete2,'k.',label='blanco+2filet.')
xlabel('Longitud de onda (nm)');ylabel('Reflectancia');title('Papel blanco');xlim(425,900);ylim(0,);legend(l

subplot(2,2,2)
plot(lambdapapel,reflectancepapel,'ro',label='crom.')
plot(lambdapapelfilete1,reflectancepapelfilete1,'r+',label='crom.+1filet.')
plot(lambdapapelfilete2,reflectancepapelfilete2,'r.',label='crom.+2filet.')
xlabel('Longitud de onda (nm)');ylabel('Reflectancia');title('Papel tintado');xlim(425,900);ylim(0,)
legend(loc='upper center')

subplot(2,2,3)
plot(lambdapapelfilete2,reflectancepapelfilete2,'r.',label='crom.+2filet.')
plot(lambdablancofilete2,reflectanceblancofilete2,'k.',label='blanco+2filet.')
xlabel('Longitud de onda (nm)');ylabel('Reflectancia');title('2 filetes');xlim(425,900);ylim(0,);legend(loc=
```

Ejercicio 4

Análisis de los espectros de reflectancia.

A la vista de los resultados, indicar qué número de láminas de ternera debemos colocar para perder la información del cromóforo (papel tintado). Justificar la respuesta. Explicar cómo podría estimarse de forma cualitativa la profundidad de penetración de la radiación. Escribir en esta celda la respuesta.

1.2 Tarea 2. Determinar el grado de oxigenación de la sangre

T2.1. Espectro de reflectancia difusa medido en el laboratorio y su ajuste

Representación gráfica del espectro de reflectancia medido.

Representar uno de los espectros de reflectancia experimentales medidos en su caso particular en el rango de longitudes de onda de 450 a 800 nm.

En la siguiente celda de código escribir el nombre del fichero de datos correspondiente. Dicho fichero habrá sido importado a la nube previamente en el directorio del alumno.

```
In []: # MODIFICAR SOLO EL NOMBRE DEL FICHERO EN ESTA CELDA Y EJECUTAR
       filenameRefDif = 'MRDcuello2.txt'

In []: # NO TOCAR, SOLO EJECUTAR

       import StringIO
       lambda_start = 450
       lambda_end = 800
       s = open(filenameRefDif).read().replace(',','.')
       datos = loadtxt(StringIO.StringIO(s))

       lambdaRefDif = datos[:,0]
       RefDif = datos[:,1]/100
       ind_start = [i for (i, val) in enumerate(lambdaRefDif) if val==lambda_start]
       ind_end = [i for (i, val) in enumerate(lambdaRefDif) if val==lambda_end]
       lambdaRefDif = lambdaRefDif[ind_start[0]:ind_end[0]+1]
       RefDif = RefDif[ind_start[0]:ind_end[0]+1]

       plot(lambdaRefDif,RefDif,'ok')
       xlabel('$\lambda$ (nm)')
       ylabel('Reflectancia Difusa')
```

Ajuste de los datos a la reflectancia teórica.

Vamos a ajustar los datos experimentales a la reflectancia teórica calculada con la ecuación de difusión:

$$R_d = Cte\ T_{epi}\ \left[\frac{z_0\,(1+\mu_{eff}\rho_1)}{4\pi\rho_1^3}e^{-\mu_{eff}\rho_1} + \frac{(z_0+4AD)\,(1+\mu_{eff}\rho_2)}{4\pi\rho_2^3}e^{-\mu_{eff}\rho_2}\right]$$

donde Cte es una constante global de normalización. $\mu_{eff} = \sqrt{3\mu_a(\mu_a+\mu_s')}$ es el coeficiente de extinción efectivo de la radiación al propagarse en el tejido, donde μ_s' es el coeficiente de scattering reducido, y μ_a es el coeficiente de absorción (todos en cm^{-1}). $D = 1/(3(\mu_a+\mu_s'))$ es el coeficiente de difusión. $z_0 = 1/(\mu_a+\mu_s')$ es la longitud de penetración de la radiación en el tejido, que es considerada en el modelo de difusión como la distancia en el interior del tejido a la cual se situa la fuente de luz. Sabiendo que la distancia entre las fibras de iluminación y detección es r (en cm), las distancias entre el detector y las dos fuentes, la que se coloca en el interior del tejido y la imaginaria negativa que se coloca fuera para ajustar la condición de contorno y extender el problema a un medio infinito, son:

$$\rho_1 = \sqrt{z_0^2 + r^2},$$

$$\rho_2 = \sqrt{(z_0+4AD)^2 + r^2}.$$

A está relacionado con el cambio de índice de refracción en la frontera, entre el tejido y el medio externo: $A = (1 + R_{eff})/(1 - R_{eff})$, con $R_{eff} = -1.440/n_r^2 + 0.71/n_r + 0.668 + 0.0636 n_r$, siendo n_r el índice de refracción del tejido normalizado por el índice del medio externo. T_{epi} es la transmitancia de la epidermis que tiene en cuenta la absorción de la melanina

$$T_{epi} = e^{-Mel\ \mu_{a\ mel}\ L_{epi}}$$

donde Mel es el contenido de melanina en la epidermis, $L_{epi} = 0.006 * 2$ cm es el camino recorrido por la luz en la epidermis al entrar y salir de ella (considerando 60 μm de epidermis), y por último $\mu_{a\ mel}$ es el coeficiente de absorción de la melanina (en cm^{-1}):

$$\mu_{a\ mel}(cm^{-1}) = \frac{6.6 \times 10^{11}}{[\lambda(nm)]^{3.33}}$$

El coeficiente de scattering reducido $\mu'_s = a(\mu'_{s\ Mie} + \mu'_{s\ Ray.})$ es una combinación de scattering Mie y Rayleigh (en cm^{-1}):

$$\mu'_{s\ Mie}(cm^{-1}) = 4.59 \times 10^3\ [\lambda(nm)]^{-0.913}$$
$$\mu'_{s\ Ray.}(cm^{-1}) = 1.74 \times 10^{12}\ [\lambda(nm)]^{-4}$$

mientras que a es una constante que varía la amplitud del scattering.

El coeficiente de absorción (en cm^{-1}) es debido principalmente a la hemoglobina y en menor media al agua:

$$\mu_a = B\ (S\ \mu_{a\ oxi} + (1 - S)\ \mu_{a\ desoxi}) + W\ \mu_{a\ water}$$

donde B es la fracción de volumen de la sangre en el tejido, S es el grado de oxigenación de la sangre, $\mu_{a\ oxi}$ y $\mu_{a\ desoxi}$ son los coeficientes de absorción de la oxyhemoglobina y desoxyhemoglobina en la sangre, respectivamente, W es la fracción de volumen de agua en el tejido y $\mu_{a\ water}$ el coeficiente de absorción del agua.

En la siguiente celda de código vamos a representar en escala semilogarítmica los coeficientes de absorción (hemoglobina, agua y melanina) y de scattering (Mie y Rayleigh) empleados.

In []:
```
# NO TOCAR, SOLO EJECUTAR

# Reduced scattering coefficients Mie and Rayleigh, cm^-1
Mie = 4.59e3*lambdaRefDif**-0.913
Ray = 1.74e12*lambdaRefDif**-4
# Absorption coefficient of interior of melanosome, cm^-1
muamel = 6.6e11*lambdaRefDif**-3.33
# Absorption coefficient of hemoglobin and water, cm^-1
datos = loadtxt('oxihemoglobina450-800nm.DAT')
muaoxy = datos[:,1];
datos = loadtxt('hemoglobina450-800nm.DAT')
muadeoxy = datos[:,1]
datos = loadtxt('agua450-800nm.DAT')
muawater = datos[:,1]
fig = figure(figsize=(13,6))
semilogy(lambdaRefDif,Mie,'c',label='$\mu_{s \; Mie}^{\prime}$')
semilogy(lambdaRefDif,Ray,'y',label='$\mu_{s \; Ray.}^{\prime}$')
semilogy(lambdaRefDif,muamel,'k',label='$\mu_{a \; mel}$')
semilogy(lambdaRefDif,muaoxy,'r',label='$\mu_{a \; oxy}$')
semilogy(lambdaRefDif,muadeoxy,'g',label='$\mu_{a \; deoxy}$')
```

```
semilogy(lambdaRefDif,muawater,'b',label='$\mu_{a \; water}$')
xlabel('$\lambda$ (nm)')
ylabel('Coeficientes de scatering y absorcion (cm$^{-1}$)')
legend(loc='lower left')
```

Para realizar el ajuste de la curva experimental a la función teórica de la reflectancia difusa deducida a partir de la ecuación de difusión necesitamos definir en la siguiente celda de código el valor de varios parámetros:

- **r**: distancia (cm) entre la fibra que ilumina y la fibra que recoge la luz difundida en el medio (comprobar el tamaño de las fibras que contiene la sonda)
- **W**: fracción de volumen de agua en los tejidos. El valor típico del contenido de agua en los tejidos es de 0.65
- **n_r**: índice de refracción del tejido / índice del medio externo. El índice del tejido puede ser 1.4.

El resto de parámetros: **B**, **S**, **Mel**, **a**, y **Cte** son usados como parámetros de ajuste, es decir, el ajuste de los datos experimentales a la función teórica se realizará variando dichos parámetros hasta conseguir que la curva teórica se aproxime lo máximo posible a la curva experimental. Así obtendremos los valores finales de dichos parámetros. Para comenzar el ajuste necesitamos definir los valores iniciales de los parámetros de ajuste:

- **B**: fracción de volumen de sangre en el tejido. El rango de valores suele ser entre 0.001 (0.1%) y 0.07 (7%).
- **S**: grado de oxigenación. Fracción de sangre oxigenada respecto del total. Valor entre 0 y 1.
- **Mel**: contenido de melanina en la epidermis. Es una fracción de volumen, el rango típico puede ir desde 0.01 (1%) en pieles claras hasta más de 0.2 (20%) en pieles oscuras.
- **a**: constante que varía la amplitud total del scattering. Vamos a empezar con un valor de 1.
- **Cte**: constante global de la reflectancia difusa teórica. Vamos a empezar con un valor de 0.1.

Sabiendo que el grado de oxigenación de la sangre viene determinado fundamentalmente por la profundidad de los picos de absorción en torno a 542 nm y 576 nm, nos limitaremos a ajustar la parte central del espectro para obtener el grado de oxigenación de la sangre. En la siguiente celda definimos por tanto nuevas variables seleccionando de las anteriores los valores correspondientes a este rango (510-610nm), y que se identifican por el sufijo `_ajuste`. **Serán por tanto estas variables las que tengamos que introducir en la función `curve_fit` a la hora de realizar el ajuste**.

In []:
```
# MODIFICAR CELDA Y EJECUTAR

r = 0.04    # cm, separación fuente- detector de fibra
W = 0.65  #  Contenido en agua del téjido
n_r = 1.4    # índice de refracción del tejido dividido por el del medio externo

# Valores iniciales de los parámeteros
B_ini = 0.01    # Fracción de volumen en sangre
S_ini = 0.5      # Grado de oxigenación
Mel_ini = 0.05     # Contenido de melanina en la epidermis
a_ini = 1.0  # Constante que escala la amplitud total del scattering del téjido.
Cte_ini = 0.1   # Constante global de la reflectancia difusa teórica

params_ini = [B_ini,S_ini,Mel_ini,a_ini,Cte_ini]

# Datos experimentales y variables auxiliares para ajustar
nini = 60
nfin = 190
```

```
lambda_ajuste = lambdaRefDif[nini:-nfin]
RefDif_ajuste = RefDif[nini:-nfin]
muaoxy_ajuste = muaoxy[nini:-nfin]
muadeoxy_ajuste = muadeoxy[nini:-nfin]
muawater_ajuste = muawater[nini:-nfin]
muamel_ajuste = muamel[nini:-nfin]
Mie_ajuste = Mie[nini:-nfin]
Ray_ajuste = Ray[nini:-nfin]
```

En la siguiente celda definimos la función modelo que reproduce la expresión

$$R_d = Cte\ T_{epi}\ \left[\frac{z_0\,(1+\mu_{eff}\rho_1)}{4\pi\rho_1^3}e^{-\mu_{eff}\rho_1} + \frac{(z_0+4AD)\,(1+\mu_{eff}\rho_2)}{4\pi\rho_2^3}e^{-\mu_{eff}\rho_2}\right]$$

Debido a que es algo más compleja que otras utilizadas en las prácticas, esta función se proporciona, y tendrá que ser utilizada en el ajuste con su nombre: `funcion_modelo`. También se importa la función `curve_fit` para poder ser utilizada en la resolución del Ejercicio 5.

In []:
```
# NO TOCAR, SOLO EJECUTAR

Lepi = 0.0060*2 # cm, distancia recorrida por la luz dentro de la epidermis

# Condición de frontera: reflexión interna (fórmulas de Fresnel)
R_eff = 0.668 + 0.0636*n_r + 0.710/n_r - 1.440/n_r**2;
A = (1 + R_eff)/(1 - R_eff)

def fun_modelo(lambdas,B,S,Mel,a,Cte):
    musp = a*(Mie_ajuste + Ray_ajuste)
    Tepi = exp(-Mel*muamel_ajuste*Lepi)
    mua  = B*(S*muaoxy_ajuste + (1.0-S)*muadeoxy_ajuste) + W*muawater_ajuste
    mueff = sqrt(3*mua*(mua+musp))
    D=1.0/(3*(mua+musp))
    z0 = 1.0/(mua + musp)
    r1 = sqrt(z0**2 + r**2)
    r2 = sqrt((z0 + 4*A*D)**2 + r**2)
    if((Mel>0.0)):
        RD = Cte*Tepi*(((z0*(1+mueff*r1))/(4*pi*r1**3))*exp(-mueff*r1) +
                       ((z0+4*A*D)*(1+mueff*r2)/(4*pi*r2**3))*exp(-mueff*r2))
    else:
        RD = 1e7*ones(shape(lambdas))
    return RD

from scipy.optimize import curve_fit
```

Ejercicio 5

Ajuste de los datos medidos al modelo teórico

En la celda anterior hemos definido la función modelo de la reflectancia difusa (`fun_modelo`). También hemos cargado la función de ajuste `curve_fit`. Utilizando los parámetros iniciales introducidos anteriormente, crea una nueva celda de código en donde,

- Realizar el ajuste de los datos experimentales utilizando esta función modelo (Nota: utilizar para ajustar las variables `lambda_ajuste` y `RefDif_ajuste`).
- Mostrar los valores de los parámetros óptimos.

- Mostrar en una gráfica el modelo teórico superpuesto a los datos experimentales

Ejercicio 6

Análisis del ajuste

¿Son razonables los valores de los parámetros de ajuste obtenidos? comentar los resultados.

T2.2. Espectro de reflectancia difusa y grado de oxigenación de la sangre

Para analizar la influencia del grado de oxigenación de la sangre (parámetro **S**) en la reflectancia difusa vamos a representar la función teórica de la reflectancia difusa **Rd** empleando los parámetros del ajuste anterior junto con otra curva en la cual podemos cambiar el valor de **S** dinámicamente.

In []:
```
from IPython.html.widgets import interact,fixed

def fun_modelo_total(lambdas,B,S,Mel,a,Cte):
    musp = a*(Mie + Ray)
    Tepi = exp(-Mel*muamel*Lepi)
    mua  = B*(S*muaoxy + (1.0-S)*muadeoxy) + W*muawater
    mueff = sqrt(3*mua*(mua+musp))
    D=1.0/(3*(mua+musp))
    z0 = 1.0/(mua + musp)
    r1 = sqrt(z0**2 + r**2)
    r2 = sqrt((z0 + 4*A*D)**2 + r**2)
    RD = Cte*Tepi*(((z0*(1+mueff*r1))/(4*pi*r1**3))*exp(-mueff*r1) +
                   ((z0+4*A*D)*(1+mueff*r2)/(4*pi*r2**3))*exp(-mueff*r2))
    return RD

def ajuste_manual(lambdas,B,S,Mel,a,Cte):
    RD_orig = fun_modelo_total(lambdas,B_opt,S_opt,Mel_opt,a_opt,Cte_opt)
    RD_nuevo = fun_modelo_total(lambdas,B,S,Mel,a,Cte)
    plot(lambdas,RD_orig,color='blue',linewidth=2,label='S original');
    plot(lambdas,RD_nuevo,color='r',linewidth=2,label='S nuevo');
    legend(loc=1)

interact(ajuste_manual,lambdas=fixed(lambdaRefDif),
         B=fixed(B_opt),S=(0.2,1,0.05),Mel=fixed(Mel_opt),
         a=fixed(a_opt),Cte=fixed(Cte_opt));
```

Ejercicio 7

Interpretar el resultado a la vista de la gráfica. Escribir en esta celda la respuesta.

T2.3. Espectro de reflectancia difusa y amplitud del scattering

Para analizar la influencia de la constante que varía la amplitud total del scattering (parámetro **a**) en la reflectancia difusa vamos a representar la función teórica de la reflectancia difusa **Rd** empleando los parámetros del ajuste junto con otra curva en la cual cambiamos el valor de **a** dinámicamente.

In []:
```
from IPython.html.widgets import interact,fixed

def fun_modelo_total(lambdas,B,S,Mel,a,Cte):
    musp = a*(Mie + Ray)
```

```
    Tepi = exp(-Mel*muamel*Lepi)
    mua  = B*(S*muaoxy + (1.0-S)*muadeoxy) + W*muawater
    mueff = sqrt(3*mua*(mua+musp))
    D=1.0/(3*(mua+musp))
    z0 = 1.0/(mua + musp)
    r1 = sqrt(z0**2 + r**2)
    r2 = sqrt((z0 + 4*A*D)**2 + r**2)
    RD = Cte*Tepi*(((z0*(1+mueff*r1))/(4*pi*r1**3))*exp(-mueff*r1) +
                   ((z0+4*A*D)*(1+mueff*r2)/(4*pi*r2**3))*exp(-mueff*r2))
    return RD

def ajuste_manual(lambdas,B,S,Mel,a,Cte):
    RD_orig = fun_modelo_total(lambdas,B_opt,S_opt,Mel_opt,a_opt,Cte_opt)
    RD_nuevo = fun_modelo_total(lambdas,B,S,Mel,a,Cte)
    plot(lambdas,RD_orig,color='blue',linewidth=2,label='a original');
    plot(lambdas,RD_nuevo,color='r',linewidth=2,label='a nuevo');
    legend(loc=1)

interact(ajuste_manual,lambdas=fixed(lambdaRefDif),
         B=fixed(B_opt),S=fixed(S_opt),Mel=fixed(Mel_opt),
         a=(0.4,1.2,0.1),Cte=fixed(Cte_opt));
```

Ejercicio 8

Interpretar el resultado a la vista de la gráfica. Escribir en esta celda la respuesta.

T2.4. Espectro de reflectancia difusa y contenido de melanina

Para analizar la influencia del contenido de melanina en la epidermis (parámetro **Mel**) en la reflectancia difusa vamos a representar la función teórica de la reflectancia difusa **Rd** empleando los parámetros del ajuste junto con otra curva en la cual cambiamos el valor de **Mel** dinámicamente.

In []:
```
from IPython.html.widgets import interact,fixed

def fun_modelo_total(lambdas,B,S,Mel,a,Cte):
    musp = a*(Mie + Ray)
    Tepi = exp(-Mel*muamel*Lepi)
    mua  = B*(S*muaoxy + (1.0-S)*muadeoxy) + W*muawater
    mueff = sqrt(3*mua*(mua+musp))
    D=1.0/(3*(mua+musp))
    z0 = 1.0/(mua + musp)
    r1 = sqrt(z0**2 + r**2)
    r2 = sqrt((z0 + 4*A*D)**2 + r**2)
    RD = Cte*Tepi*(((z0*(1+mueff*r1))/(4*pi*r1**3))*exp(-mueff*r1) +
                   ((z0+4*A*D)*(1+mueff*r2)/(4*pi*r2**3))*exp(-mueff*r2))
    return RD

def ajuste_manual(lambdas,B,S,Mel,a,Cte):
    RD_orig = fun_modelo_total(lambdas,B_opt,S_opt,Mel_opt,a_opt,Cte_opt)
    RD_nuevo = fun_modelo_total(lambdas,B,S,Mel,a,Cte)
    plot(lambdas,RD_orig,color='blue',linewidth=2,label='Mel original');
    plot(lambdas,RD_nuevo,color='r',linewidth=2,label='Mel nuevo');
    legend(loc=1)

interact(ajuste_manual,lambdas=fixed(lambdaRefDif),
```

```
B=fixed(B_opt),S=fixed(S_opt),Mel=(0.01,0.2,0.02),
a=fixed(a_opt),Cte=fixed(Cte_opt));
```

Ejercicio 9

Interpretar el resultado a la vista de la gráfica. Escribir en esta celda la respuesta.

Guión Práctica 4

Sonia Melle Hernández

1 Microscopía de Fluorescencia

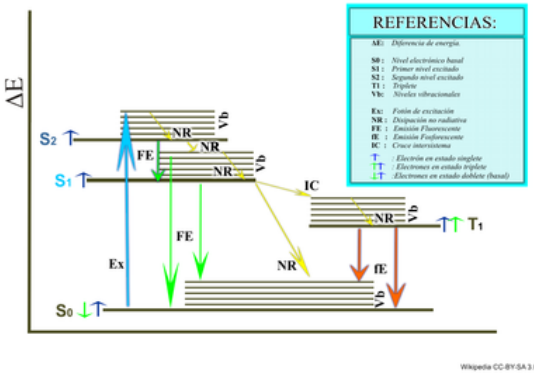
1.1 Objetivos

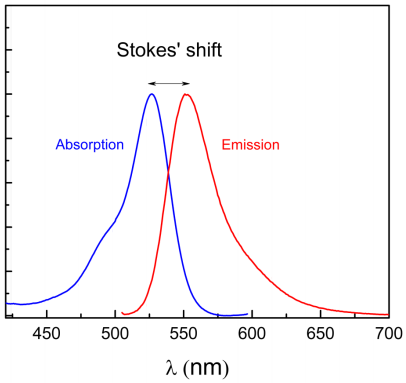
- Escoger y caracterizar la fuente y los filtros de excitación, emisión y dicroico para construir un microscopio que permita ver la fluorescencia de un determinado fluoróforo.
- Adaptar un microscopio convencional comercial para que funcione como microscopio de fluorescencia para ese fluoróforo.
- Tomar imágenes de fluorescencia y determinar el tamaño de las estructuras observadas.

¿Qué es la Fluorescencia?

La absorción de radiación por parte de un medio involucra una transición entre dos estados de energía del átomo o molécula que interacciona con la luz. Para el caso de absorción por parte de una molécula, los niveles involucrados corresponderán a estados electrónicos (cuando la frecuencia de la luz absorbida corresponda al visible y el ultravioleta), vibracionales (si la frecuencia de la radiación absorbida corresponde al infrarrojo medio y cercano) o rotacionales (si corresponde al infrarrojo lejano y microondas). Una vez el átomo o molécula se excita a un nivel de energía superior, puede liberar energía por medio de la emisión de luz. Se denomina luminiscencia a la emisión de luz por parte del medio al desexcitarse desde un nivel de energía electrónico. En la siguiente figura se presenta un esquema de los procesos que dan lugar a la fluorescencia por medio de un diagrama de niveles. En un primer momento, la molécula se excita por interacción con la luz incidente hasta uno de los niveles vibracionales de los estados electrónicos superiores S_1, S_2 en la figura. Desde ese nivel, la molécula experimenta normalmente una desexcitación por medio de procesos no radiativos (es decir, que no conllevan emisión de fotones) hasta el nivel vibracional más bajo del primer nivel electrónico excitado S_1. Este proceso se produce muy rápidamente, con tiempos menores del ns. Desde este estado, la molécula puede perder energía emitiendo luz que le lleva a uno de los niveles vibraciones del estado electrónico fundamental. Esta emisión es la que se conoce como fluorescencia y se produce en un tiempo típico de ns. Se denomina a su vez como fluoróforo una molécula capaz de emitir luz por este proceso. Como se puede observar, en el proceso de fluorescencia intervienen estados electrónicos más cercanos en energía que los que conectan la absorción de la luz incidente. Por tanto, la fluorescencia tendrá una frecuencia menor que la radiación que ha excitado la molécula. Se denomina desplazamiento de Stokes a la diferencia en frecuencias entre los máximos de las bandas de emisión y de absorción de la molécula.

Cuando la molécula se encuentra en el estado más bajo del primer nivel electrónico S_1, se puede llegar a dar otro proceso de transferencia de energía que no involucra la emisión de un fotón. Esta transición se denomina *Sistema Intercruzado* y se denota en la figura superior por las siglas *IC*. Consiste en la transición de la molécula a otro estado denominado Triplete (T_1 en la figura) en donde el espín del electrón cambia. La consecuencia fundamental de este cambio es que la relajación de la molécula desde el estado triplete al estado fundamental pasa a producirse en tiempos mucho más largos (del orden del milisegundo). A la emisión que se produce en este proceso de relajación se denomina *Fosforescencia* para distinguirla de la





relajación desde el estado singlete excitado que da lugar a una emisión en tiempos mucho más rápidos (y que es la que denominamos Fluorescencia).

Para medir la fluorescencia normalmente se sitúa una cámara o un fotodiodo en la dirección perpendicular a la del haz utilizado para excitar la muestra. De este modo se minimiza la contribución a la radiación capturada del haz incidente, quedando únicamente la fluorescencia. Además se utilizan filtros adecuados con el fin de eliminar completamente la radiación de otras longitudes de onda no deseadas (que pueden llegar al fotodiodo o la cámara por scattering o reflexiones).

Las medidas de fluorescencia son importantes para el estudio de tejidos biológicos y de materiales. Al medir la fluorescencia, obtenemos información de la estructura de niveles de la molécula bajo estudio y de su interacción con el medio que la rodea. Además, las imágenes de fluorescencia presentan un alto contraste, lo que permite discriminar la localización de moléculas a nivel intracelular. Por otro lado, podemos utilizar determinados fluoróforos para marcar, por ejemplo, ciertas proteínas no fluorescentes. Conociendo la fluorescencia de estos fluoróforos podemos localizar y seguir la evolución de esas proteínas.

De las características que definen la fluorescencia de una sustancia, quizás el **rendimiento cuántico** y el **tiempo de decaimiento** de la fluorescencia son las más importantes. El rendimiento cuántico es el cociente entre el número de fotones emitidos y el número de fotones absorbidos, y nos da una idea del número de moléculas que se desexcitan por emisión, dando lugar a la fluorescencia, frente al número que se desexcitan por procesos no radiativos. Estos procesos no radiativos, pueden ser, por ejemplo, colisiones con moléculas del entorno o bien transferencia de energía resonante (RET por sus siglas en inglés). En el primer caso, la molécula sufre una colisión con otra de su entorno que le lleva a perder la energía adquirida y a retornar al estado fundamental. En el caso del RET, la pérdida de energía se produce cuando el espectro de emisión del fluoróforo se solapa con el espectro de absorción de otra molécula próxima. En este caso, y sin que se produzca emisión por parte del fluoróforo, se produce una transferencia de energía entre ambas moléculas por medio de interacción eléctrica (dipolo-dipolo). El tiempo de decaimiento nos da una idea del tiempo que tiene la molécula para interaccionar con su entorno estando en el nivel excitado. Es por ello que siendo capaces de registrar la evolución en el tiempo de la señal de fluorescencia, podremos obtener información tanto de los procesos en la molécula que tienen lugar en ese tiempo como del entorno. Por ejemplo, podemos observar si la molécula rota durante el proceso de emisión.

Para saber un poco más del fenómeno de fluorescencia,

Vídeo sobre microscopía de fluorescencia

Vídeo sobre fluorescencia

Tutorial sobre la fluorescencia

Tutorial sobre los espectros de excitación y emisión

Espectros de excitación y de emisión

Usualmente la fluorescencia de una molécula se visualiza mediante los espectros de excitación y de emisión. Vamos a ver qué significa cada uno de ellos.

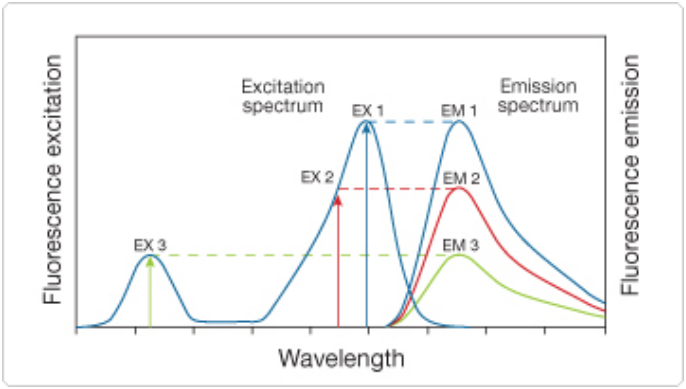


Figure 1: espectros

Espectro de excitación

Imaginemos una muestra que contiene un cierto número de moléculas fluorescentes, todas ellas iguales. Cuando iluminemos con radiación de una cierta longitud de onda excitaremos una cierta fracción de esas moléculas, mayor cuanto más eficiente sea el proceso de excitación. Si registramos la fluorescencia emitida, ésta será más intensa cuanto mayor sea el número de moléculas excitadas, es decir, cuanto más eficiente sea el proceso de excitación. Esta medida es lo que nos da el espectro de excitación. Variamos de forma contínua la longitud de onda de la radiación que ilumina la muestra, y medimos cómo cambia la señal de fluorescencia a una cierta longitud de onda fija. De este modo obtenemos una curva de excitación frente a longitud de onda de la radiación incidente.

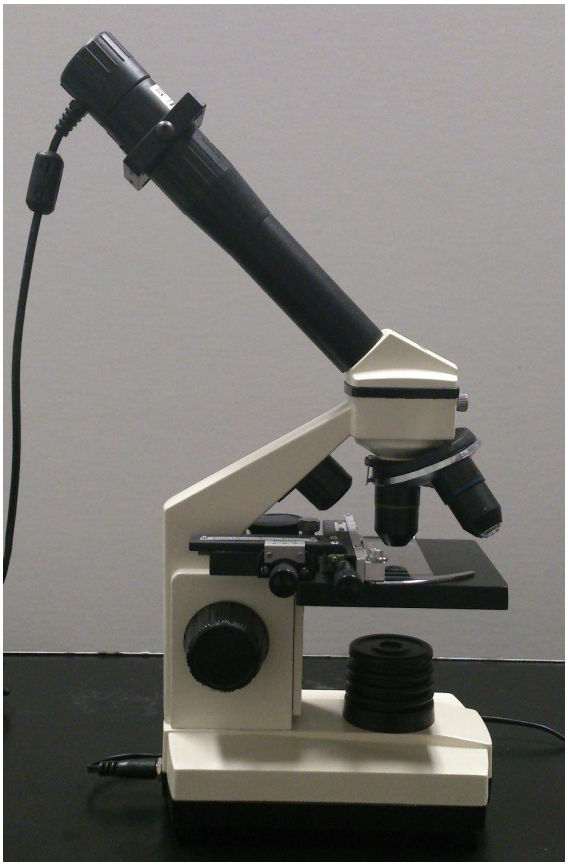
Espectro de emisión

Para esta medida se fija la longitud de onda de la radiación incidente en la muestra, y se mide cómo cambia la fluorescencia con la longitud de onda. Es necesario indicar que la forma del espectro de emisión no cambia al variar la longitud de onda de la radiación incidente. Este cambio únicamente da lugar a un mayor o menor número de moléculas excitadas, lo que cambia la amplitud del espectro de emisión, pero no su forma. El espectro de emisión se extiende por un cierto rango de longitudes de onda debido a la presencia de niveles vibracionales en el nivel electrónico fundamental. Al desexcitarse la molécula desde el nivel electrónico S_1, puede hacerlo a cualquiera de los niveles vibracionales del estado fundamental, cambiando por tanto la longitud de onda de emisión.

2 Realización experimental

En esta práctica vamos a adaptar un microscopio convencional (figura de la izquierda) para que funcione como microscopio de fluorescencia (figura de la derecha) para un determinado fluoróforo. Para ello precisamos incorporar en el microscopio convencional tanto la fuente de excitación como los filtros de excitación, dicroico y de emisión. El alumno dispone de una serie de piezas optomecánicas (cubo portafiltros, barras, tubos,...) así como filtros, fuentes LED y lentes, que le permitirán ensamblar el dispositivo.

En la siguiente figura se muestra el esquema de un microscopio de fluorescencia indicando sus principales componentes. Este microscopio permite observar la emisión fluorescente de unas determinadas moléculas. Para ello tendremos que iluminar la muestra con radiación de una cierta longitud de onda (azul, en el ejemplo), y a su vez, ser capaces de registrar la emisión de fluorescencia a otra longitud de onda (verde



en el ejemplo). La emisión fluorescente es una señal muy débil, debido a dos causas. La primera es que normalmente la concentración de fluoróforos no es muy alta, con lo que únicamente una fracción de la radiación incidente logra excitar las moléculas de interés. Y la segunda causa es que las pérdidas de energía por procesos no radiativos, así como la absorción de la emisión por el entorno, hacen que la señal fluorescente que llega al detector sea aún más débil. La señal de fluorescencia puede ser de 3 a 6 órdenes de magnitud inferior a la de iluminación. Por todo ello es necesario disponer de filtros adecuados y de una iluminación intensa para obtener imágenes nítidas de fluorescencia. Normalmente para la excitación del fluoróforo se dispone de una fuente junto con un filtro de excitación. Este filtro se utiliza para seleccionar del espectro de emisión de la fuente las longitudes de onda adecuadas para excitar la molécula de interés. Para realizar la observación, se dispone de otro filtro denominado filtro de emisión. Su función es eliminar la radiación de longitudes de onda distintas a la que queremos registrar y que nos ocultarían la señal fluorescente. Por ejemplo, aunque la mayor parte de la radiación utilizada para excitar la muestra pasa a través de ella, la señal reflejada o dispersada puede ser aún mucho mayor que la emisión fluorescente. El filtro de emisión se encarga de limpiar la radiación que llega al detector para maximizar el contraste de la imagen. Para combinar la excitación y la emisión, se coloca finalmente un filtro dicroico, el cual se escoge para reflejar la longitud de onda de la luz que usamos para excitar la muestra, mientras que transmite la longitud de onda que queremos observar (conviene recordar en este punto que la fluorescencia se emite en una longitud de onda mayor que la de absorción). Aunque la presencia del filtro dicroico ya elimina gran parte de la radiación que puede ocultar la fluorescencia, es necesario el filtro de emisión para aislar aún más esta débil emisión.

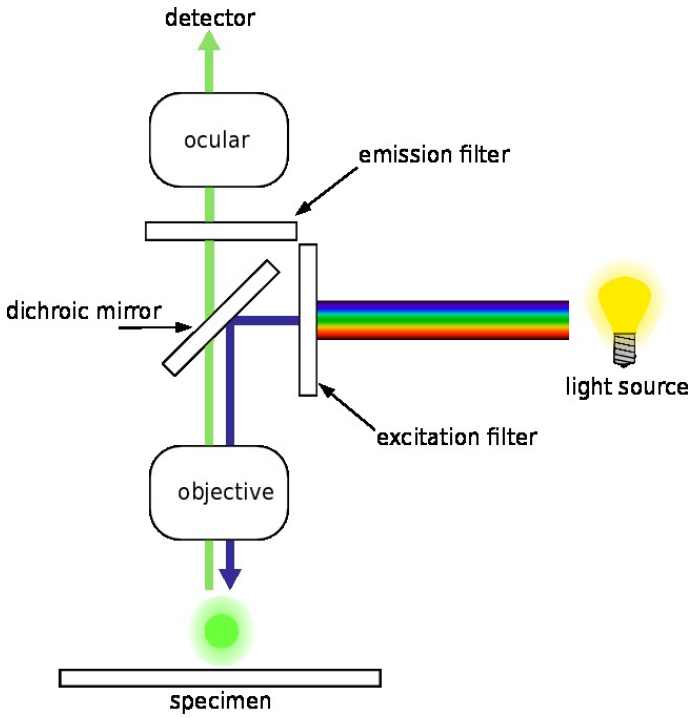


Figure 2: Microscopio de Fluorescencia

Parte 1: Elección y caracterización espectral de la fuente de iluminación y los filtros

En esta parte de la práctica vamos a escoger los filtros adecuados para excitar y registrar la emisión del fluoróforo asignado a cada grupo. Disponemos de tres fluoróforos distintos:

- *Partículas Nile Red*. Son partículas micrométricas de poliestireno que contienen un fluoróforo que fluoresce en rojo (con su máximo en λ=636 nm) al ser excitado con radiación verde. Estan comercializadas por la empresa *Invitrogen* (referencia F8825) y vienen disueltas en agua destilada. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa Thorlabs cuyo máximo de emisión está en λ=530 nm (Modelo: M530L3-Green).
- *Proflavina* en polvo disuelta en agua destilada. Es comercializada por la empresa *Sigma-Aldrich* (referencia 131105-5G). Fluoresce en verde (con su máximo en λ=512 nm) cuando se excita con azul. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa Thorlabs cuyo máximo de emisión está en λ=455 nm (Modelo: M455L2-Royal Blue).
- *Protoporfirina IX* (PpIX) en polvo comercializada por la empresa *Sigma-Aldrich* (referencia P8293-1G). Para su buena dispersión debe ser disuelta en una mezcla de N,N-dimetilformamida y metanol al 50%. Fluoresce en rojo (con su máximo en λ=633 nm) cuando se excita con UV-azul. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa *LedEngin* cuyo máximo de emisión está en λ=390 nm (Modelo: LZ1-10UA00).

Tanto la fuente de excitación como los filtros que se emplearán para cada fluoróforo **habrán sido elegidos por el alumno en la tarea previa a la práctica**. En la mesa de cada grupo encontramos el siguiente

material:

- Filtros
- Cubo portafiltros en cuyo interior se encuentra alojado el filtro dicroico
- Fuente LED
- Fuente halógena
- Espectrómetro
- Lente de focal 35 mm
- Red de Ronchi
- Microscopio
- Piezas optomecánicas (pieza de adaptación microscopio-cubo, 4 barras, 1 cilindro, 1 tubo, 1 diafragma)
- Jeringuilla
- Portaobjetos
- Fluoróforo
- Llave Allen

1) Medimos el espectro de la fuente LED utilizando el dispositivo cuyo esquema se muestra a continuación:

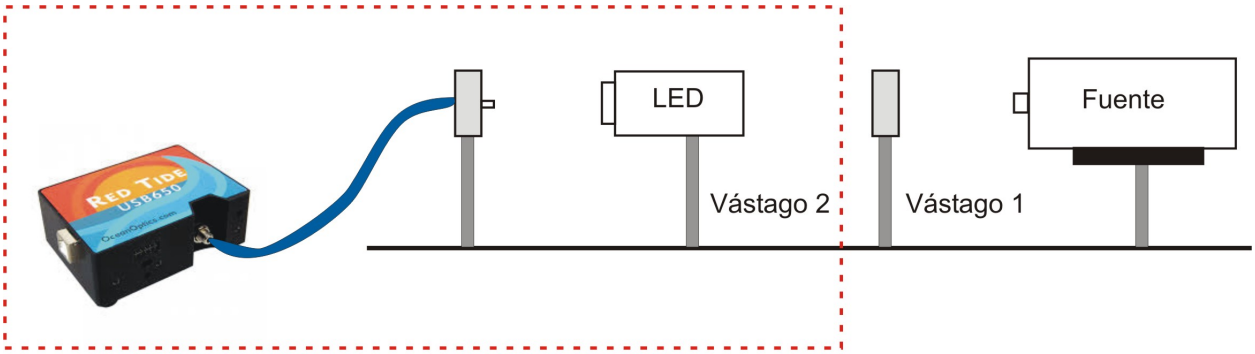


Figure 3: Setup 1

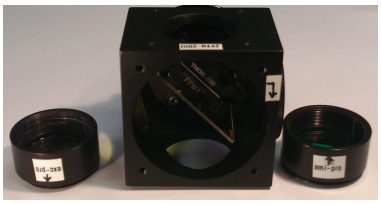
- Montamos el LED en el vástago 2 de forma que ilumine directamente la fibra que lleva la radiación al espectrómetro.
- Encedemos el LED girando la botón que se encuentra en la fuente de alimentación desde la posición cero en sentido antihorario.
- Abrimos el programa que controla el espectrómetro *SpectraSuite*.
- Modificamos el tiempo de integración para conseguir ver el espectro de emisión del LED en pantalla con suficiente intensidad.
- Salvamos dicho espectro en formato *lengueta delimitada ninguna* con nombre: "LED.txt"
- Apagamos el LED.
- Retiramos el LED del vástago 2.

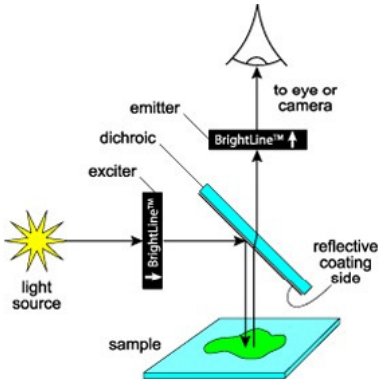
2) Medimos el espectro de transmitancia de los tres filtros utilizando el siguiente montaje:

El filtro de excitación y el de emisión están montados en una pieza cilíndrica con rosca (ver figura). En estos filtros indicamos con una flecha la dirección en la que debe propagarse la luz incidente para su correcto funcionamiento.

El filtro dicroico está colocado en el interior de un cubo portafiltros formando un ángulo de 45º respecto de las caras del mismo. Esta es la posición en la que lo utilizaremos en el microscopio. Al igual que los otros

filtros, el filtro dicroico presenta una flecha que indica la dirección de propagación de la luz (ver figura).





Para medir el espectro de transmisión de los distintos filtros vamos a iluminarlos con una lámpara halógena (LS1 o HL-2000) y utilizaremos el sistema experimental que aparece en el siguiente esquema (Figura A para medir la transmisión del filtro dicroico y Figura B para medir la transmisión de los filtros de excitación y emisión) . Para medir su transmitancia:

- Encedemos la fuente halógena con cuidado de no desplazarla pues el sistema se encuentra alineado.
- Medimos la irrandiancia incidente en el filtro (irradiancia de la fuente) con la lámpara halógena encendida. Escoger un *Tiempo de integración* y un *Boxcar width* adecuado.
- Seleccionamos en el *SpectaSuite* la opción *Fichero-> Nueva-> Medida de transmitancia–> Next*. Introducimos en la pantalla los valores del *Tiempo de integración* y *Boxcar width* elegidos con anterioridad. Presionamos *Next*.
- Presionamos sobre la bombilla amarilla para que el software guarde en memoria el espectro de la fuente y presionamos *Next*.
- A continuación medimos la irradiancia de la luz del laboratorio que llega a la fibra pero no proviene de la fuente de iluminación. Para ello interponemos un papel entre la lámpara y la fibra óptica y presionamos sobre la bombilla gris. El software guardará en memoria el espectro de oscuridad. Presionamos *Finish*. Debemos asegurarmos de que obtenemos una transmitancia del 100% para todas la longitudes de onda.

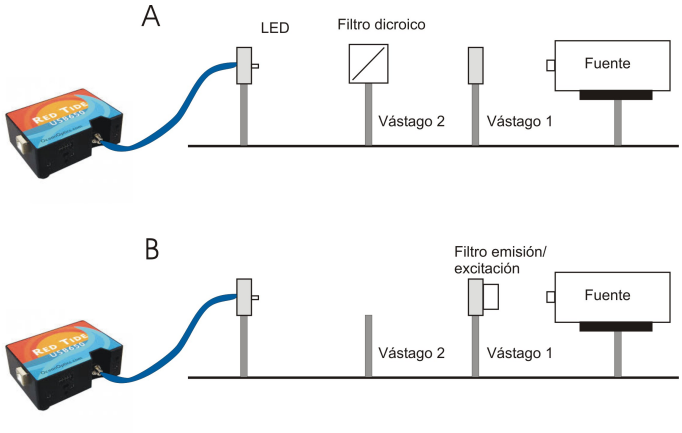


Figure 4: Setup 2

- Colocamos el filtro dicroico en el lugar donde antes estaba el LED, sobre el vástago 2 (ver figura A). Tener en cuenta la correcta orientación del filtro. Salvamos el espectro de transmisión del filtro dicroico

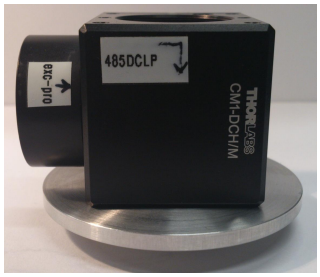
en formato *lengueta delimitada ninguna*. Fichero: "dicroico.txt"

- Quitamos el filtro dicroico del vástago 2 y enroscamos el filtro de excitación sobre el cuadradito del vástago 1 (ver figura B). Tener en cuenta la correcta orientación del filtro. Salvamos el espectro de transmisión del filtro de excitación en formato *lengueta delimitada ninguna*. Fichero: "excitacion.txt"
- Quitamos el filtro de exitación del vástago 1 y enroscamos el filtro de emisión sobre el mismo vástago. Tener en cuenta la orientación del filtro. Salvamos el espectro de transmisión del filtro de emisión en formato *lengueta delimitada ninguna*. Fichero: "emision.txt"
- Apagamos la fuente halógena y cerramos el *SpectraSuite*.

Parte 2: Montaje del microscopio de fluorescencia

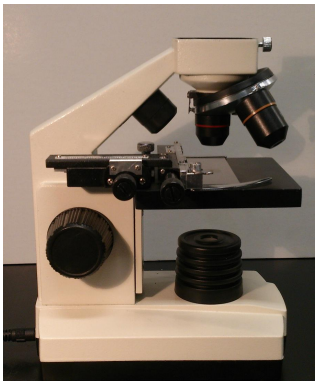
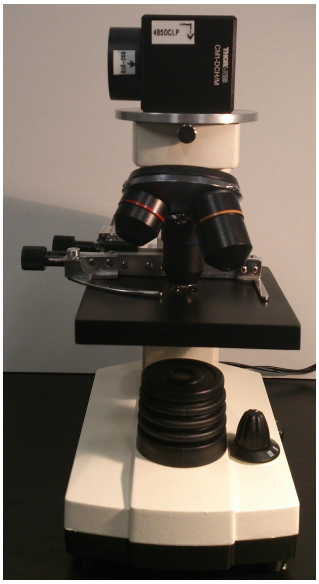
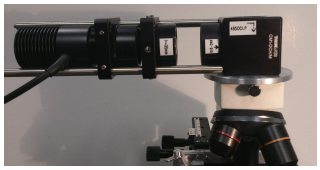
Vamos a adaptar el microscopio para conseguir ver la fluorescencia del fluoróforo seleccionado. Incorporaremos la fuente LED y los filtros seleccionados en el microscopio comercial. Colocaremos el brazo de iluminación en la dirección horizontal y el de detección en la vertical.

1) Enroscamos el filtro de excitación en el cubo portafiltros que contiene el filtro dicroico (ver figuras izquierda y centro). La flecha sobre el filtro de excitación indica la dirección de incidencia de la luz procedente del LED. El filtro dicroico debe estar colocado de forma que la luz que provenga del LED se refleje en él y se dirija hacia la muestra situada sobre la platina del microscopio. La flecha sobre el filtro dicroico indica la dirección de propagación que debe llevar la luz de excitación del LED. La fluorescencia emitida por la muestra atravesará el filtro dicroico y llegará a la cámara tras atravesar el filtro de emisión.
2) Colocamos el cubo portafiltros sobre la pieza de aluminio diseñada para unir el microscopio comercial al cubo portafiltros. La cara del cubo enfrentada a la cara donde alojaremos el filtro de emisión debe colocarse en contacto con la pieza de alumninio (ver figura derecha).



3) En el microscopio, retiramos la pieza donde va ensamblado el tubo que aloja la cámara. Para ello, desatornillamos el tornillo y retiramos con cuidado la pieza. En el dispositivo original del microscopio esta pieza contiene en su interior un prisma óptico que permite llevar la imagen de la muestra colocada en la platina al sensor de la cámara (figura izquierda).
4) Colocamos la pieza de aluminio sobre el microscopio fijando el tornillo de sujección (figura centro). Tapamos con una tapa o un papel negro la cara del cubo opuesta al filtro de excitación.
5) Enroscamos dos barras en el lateral del cubo donde se encuentra el filtro de excitación. En estas barras alojaremos el sistema de iluminación. Las barras deben estar una diagonalmente opuesta a la otra para asegurar una buena sujección de los componentes del sistema de iluminación.
6) En las barras del sistema de iluminación colocamos una lente de focal 35 mm. Retiramos el vástago del LED y colocamos el LED a continuación de la lente (figura derecha). La lente nos permitirá modificar el área iluminada de la muestra situada en la platina. Este área debe ser extensa y lo mas uniforme

posible. Concentraremos el haz mas o menos según la intensidad que necesitemos para detectar la fluorescencia.

7) En la parte superior del cubo, colocamos el filtro de emisión. Enroscamos dos barras en la parte superior del cubo para alojar el sistema de detección (figura izquierda).

8) En las barras del sistema de detección colocamos un tubo para evitar que penetre luz espúrea a la cámara (figura centro). Colocamos la cámara (figura derecha). Una vez ensamblado el sistema procedemos a enfocarlo para lo que precisamos un software que nos de la imagen que está recogiendo la cámara.



Parte 3: Toma de imágenes

Primero vamos a calibrar el aumento del microscopio. Para ello:

1) Colocamos una red de Ronchi de 40 líneas/mm sobre la platina del microscopio. NOTA: En los puestos de protoporfirina colocar la red de Ronchi sobre cuatro portaobjetos que se encuentran en la mesa. Los necesitamos porque mediremos la fluorescencia en una cubeta circular blanca situada sobre un portaobjetos de forma que la altura de ambos sistemas (cubeta y red de Ronchi) sea la misma y así podremos calibrar el tamaño de las estructuras encontradas (ver imagen). En el resto de grupos la red se colocará directamente sobre la platina.

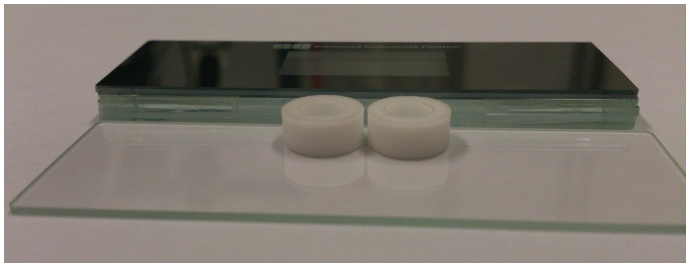


Figure 5: Cubeta

2) Abrimos el software Micam cuyo icono se encuentra en el escritorio del portátil. Seleccionamos en el menu la entrada Device–>Select Device–> Select a Video Device USB 2.0 Camera.

3) Encendemos la fuente halógena situada debajo de la platina del microscopio girando el botón situado en la parte inferior derecha desde la posición OFF a la posición I. Esto nos permitirá iluminar la red desde abajo y medir la intensidad transmitida por la misma.

4) Con el objetivo de menor aumento (4X) iluminamos la red e intentamos enfocar la imagen desplazando el botón de enfoque fino del microscopio. Si no conseguimos formar la imagen tendremos que variar la distancia entre el cubo portafiltros y la cámara hasta conseguir una imagen nítida en la pantalla del ordenador. Salvamos la imagen obtenida.

5) Incrementando los aumentos (10X y 40X) y enfocando de nuevo con el boton de enfoque fino del miscroscopio salvamos las imágenes de la red de Ronchi para 10X y 40X.

6) Retiramos la red de Ronchi y apagamos la lámpara halógena del microscopio girando el botón situado en la parte inferior derecha desde la posición I a la OFF.

Una vez montado y calibrado el aumento del microscopio de fluorescencia, podemos capturar imágenes de la emisión fluorescente de una muestra.

1) Colocamos un portaobjetos sobre la platina.

2) Encendemos el LED y ajustamos el área del portaobjetos que queremos iluminar. Colocamos un papel blanco sobre el porta. Desplazamos la lente del brazo de iluminación hasta conseguir iluminar de forma uniforme un área extensa del papel. Retiramos el papel.

3) Dependiendo del fluoróforo asignado seguir el siguiente procedimiento:

- Nile Red: Depositamos una pequeña gota de la disolución que contiene las partículas Nile Red sobre el portaobjetos. Colocamos un cubre-objetos sobre la zona del portaobjetos donde hemos depositado la gota para que las partículas se desplacen lo menos posible. Desplazamos la platina de forma que la gota ocupe sólo una porción pequeña de la imagen recogida por la cámara. Así podremos utilizar el borde de la gota o el borde del cubreobjetos para enfocar la imagen con el objetivo 4X. Con el objetivo 10X enfocar la imagen para ver las partículas. Salvaremos imágenes de las partículas esféricas con

distintos aumentos (10X y 40X) con el fin de determinar el tamaño de las mismas. Buscar una zona donde las partículas no estén aglomeradas. Controlar la intensidad del led para que la fluorescencia de las partículas no esté saturada.

- Proflavina: Inyectar proflavina en un trozo pequeño de papel depositando sobre el portaobjetos hilos del papel con proflavina y otros sin profavina. Formar imagen de algunos de los hilos del borde del papel con el aumento 4X. Comprobar que al iluminar el papel que no tiene proflavina no se produce fluorescencia. Con los objetivos 4X y 10X salvaremos imágenes del borde del papel para determinar el tamaño de las estructuras fluorescentes observadas. Controlar la intensidad del led para que la fluorescencia no esté saturada.
- Protoporfirina IX: Llenar una de las cubetas de protoporfirina dejando la otra vacía. Sobre ambas depositar una tela pues la utilizaremos para determinar el tamaño de las estructuras que la forman. Con el aumento 10X comprobar que sobre la cubeta sin protoporfirina no se pueden observar los hilos de la tela mientras que sobre la cubeta con protoporfirna si. Salvaremos imágenes de la cubeta con protoporfirina con distintos aumentos (10X y 40X). Determinaremos el tamaño de los hilos que forman la tela. Controlar la intensidad del led para que la fluorescencia de la protoporfirina no esté saturada.

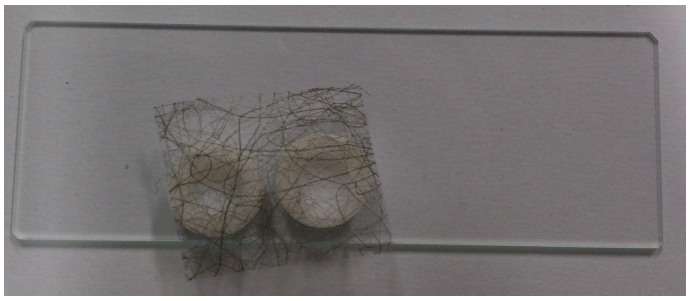


Figure 6: Cubeta con tela

Trabajo Previo Práctica 4

Sonia Melle Hernández, Oscar Gómez Calderón

1 Óptica Biomédica. Trabajo previo Práctica 4

NOMBRE Y APELLIDOS:

FECHA DE ENTREGA:

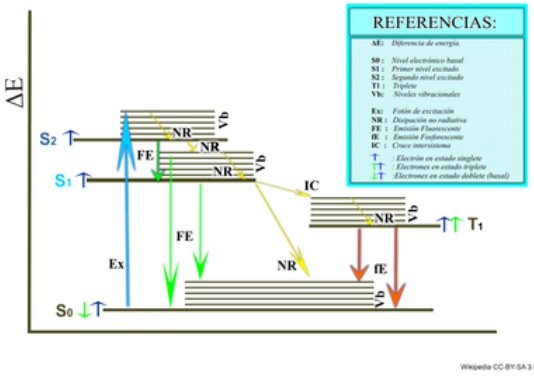
Modificar esta celda incluyendo la información solicitada

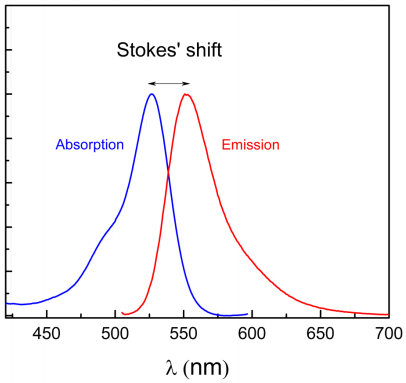
Ejecutar la siguiente celda para cargar el módulo de cálculo científico (`%pylab inline`).

```
In []: %pylab inline
```

1.1 Conceptos básicos sobre microscopía de fluorescencia

La absorción de radiación por parte de un medio involucra una transición entre dos estados de energía del átomo o molécula que interacciona con la luz. Para el caso de absorción por parte de una molécula, los niveles involucrados corresponderán a estados electrónicos (cuando la frecuencia de la luz absorbida corresponda al visible y el ultravioleta), vibracionales (si la frecuencia de la radiación absorbida corresponde al infrarrojo medio y cercano) o rotacionales (sicorresponde al infrarrojo lejano y microondas). Una vez el átomo o molécula se excita a un nivel de energía superior, puede liberar energía por medio de la emisión de luz. Se denomina luminiscencia a la emisión de luz por parte del medio al desexcitarse desde un nivel de energía electrónico. En la siguiente figura se presenta un esquema de los procesos que dan lugar a la fluorescencia por medio de un diagrama de niveles. En un primer momento, la molécula se excita por interacción con la luz incidente hasta uno de los niveles vibracionales de los estados electrónicos superiores S_1, S_2 en la figura. Desde ese nivel, la molécula experimenta normalmente una desexcitación por medio de procesos no radiativos (es decir, que no conllevan emisión de fotones) hasta el nivel vibracional más bajo del primer nivel electrónico excitado S_1. Este proceso se produce muy rápidamente, con tiempos menores del ns. Desde este estado, la molécula puede perder energía emitiendo luz que le lleva a uno de los niveles vibraciones del estado electrónico fundamental. Esta emisión es la que se conoce como fluorescencia y se produce en un tiempo típico de ns. Se denomina a su vez como fluoróforo una molécula capaz de emitir luz por este proceso. Como se puede observar, en el proceso de fluorescencia intervienen estados electrónicos más cercanos en energía que los que conectan la absorción de la luz incidente. Por tanto, la fluorescencia tendrá una frecuencia menor que la radiación que ha excitado la molécula. Se denomina desplazamiento de Stokes a la diferencia en frecuencias entre los máximos de las bandas de emisión y de absorción de la molécula.





Cuando la molécula se encuentra en el estado más bajo del primer nivel electrónico S_1, se puede llegar a dar otro proceso de transferencia de energía que no involucra la emisión de un fotón. Esta transición se denomina *Sistema Intercruzado* y se denota en la figura superior por las siglas *IC*. Consiste en la transición de la molécula a otro estado denominado Triplete (T_1 en la figura) en donde el espín del electrón cambia. La consecuencia fundamental de este cambio es que la relajación de la molécula desde el estado triplete al estado fundamental pasa a producirse en tiempos mucho más largos (del orden del milisegundo). A la emisión que se produce en este proceso de relajación se denomina *Fosforescencia* para distinguirla de la relajación desde el estado singlete excitado que da lugar a una emisión en tiempos mucho más rápidos (y que es la que denominamos Fluorescencia).

Para medir la fluorescencia normalmente se sitúa una cámara o un fotodiodo en la dirección perpendicular a la del haz utilizado para excitar la muestra. De este modo se minimiza la contribución a la radiación capturada del haz incidente, quedando únicamente la fluorescencia. Además se utilizan filtros adecuados con el fin de eliminar completamente la radiación de otras longitudes de onda no deseadas (que pueden llegar al fotodiodo o la cámara por scattering o reflexiones).

Las medidas de fluorescencia son importantes para el estudio de tejidos biológicos y de materiales. Al medir la fluorescencia, obtenemos información de la estructura de niveles de la molécula bajo estudio y de su interacción con el medio que la rodea. Además, las imágenes de fluorescencia presentan un alto contraste, lo que permite discriminar la localización de moléculas a nivel intracelular. Por otro lado, podemos utilizar determinados fluoróforos para marcar, por ejemplo, ciertas proteínas no fluorescentes. Conociendo la fluorescencia de estos fluoróforos podemos localizar y seguir la evolución de esas proteínas.

De las características que definen la fluorescencia de una sustancia, quizás el **rendimiento cuántico** y el **tiempo de decaimiento** de la fluorescencia son las más importantes. El rendimiento cuántico es el cociente entre el número de fotones emitidos y el número de fotones absorbidos, y nos da una idea del número de moléculas que se desexcitan por emisión, dando lugar a la fluorescencia, frente al número que se desexcitan por procesos no radiativos. Estos procesos no radiativos, pueden ser, por ejemplo, colisiones con moléculas del entorno o bien transferencia de energía resonante (RET por sus siglas en inglés). En el primer caso, la molécula sufre una colisión con otra de su entorno que le lleva a perder la energía adquirida y a retornar al estado fundamental. En el caso del RET, la pérdida de energía se produce cuando el espectro de emisión del fluoróforo se solapa con el espectro de absorción de otra molécula próxima. En este caso, y sin que se produzca emisión por parte del fluoróforo, se produce una transferencia de energía entre ambas moléculas por medio de interacción eléctrica (dipolo-dipolo). El tiempo de decaimiento nos da una idea del tiempo que tiene la molécula para interaccionar con su entorno estando en el nivel excitado. Es por ello que siendo capaces de registrar la evolución en el tiempo de la señal de fluorescencia, podremos obtener información tanto de los procesos en la molécula que tienen lugar en ese tiempo como del entorno. Por ejemplo, podemos observar si la molécula rota durante el proceso de emisión.

Para saber un poco más del fenómeno de fluorescencia,

Vídeo sobre microscopía de fluorescencia

Vídeo sobre fluorescencia

Tutorial sobre la fluorescencia

Tutorial sobre los espectros de excitación y emisión

Ejercicio 1

Explicar:

A) ¿qué es el espectro de excitación y el espectro de emisión de un fluoróforo?

B) ¿qué procedimiento experimental se emplea para medir cada uno de ellos?

Conteste en esta celda a estas cuestiones

Ejercicio 2

En un microscopio de fluorescencia, explicar el papel que desempeñan cada uno de los filtros empleados: filtro de excitación, dicroico, y filtro de emisión. Conteste en esta celda a esta cuestión

---

Vamos a diseñar un microscopio de fluorescencia escogiendo los filtros adecuados para excitar y registrar la emisión de distintos fluoróforos. En la práctica utilizaremos tres fluoróforos:

- *Nile Red*: partículas micrométricas fluorescentes (de la empresa Invitrogen, referencia F8825) disueltas en agua destilada. Fluorescen en rojo al ser excitadas con radiación verde. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa Thorlabs cuyo máximo de emisión está en λ=530 nm (Modelo: M530L3-Green).
- *Proflavina*: (de la empresa Sigma-Aldrich, referencia 131105-5G) disuelta en agua destilada. Fluoresce en verde (con su máximo en λ=512 nm) cuando se le excita con azul. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa Thorlabs cuyo máximo de emisión está en λ=455 nm (Modelo: M455L2-Royal Blue).
- *Protoporfirina IX*: (de la empresa Sigma-Aldrich, referencia P8293-1G) disuelta en una mezcla de N,N-dimetilformamida y etanol al 50%. Fluoresce en rojo (con su máximo en λ=633 nm) cuando se excita en UV-azul. Como fuente de excitación utilizaremos un LED de alta potencia de la empresa LedEngin cuyo máximo de emisión está en λ=390 nm (Modelo: LZ1-10UA00).

1.2 Fluoróforo Nile Red

Cargamos el fichero del espectro de excitación de las partículas *Nile Red*: **excitacion_NILE_RED.txt**. La primera columna es la longitud de onda y la segunda columna es la intensidad relativa del espectro de excitación. Igualmente cargamos el fichero del espectro de emisión de las partículas *Nile Red*, **emision_NILE_RED.txt**; la primera columna es la longitud de onda para el espectro de emisión, y la segunda columna es la intensidad relativa del espectro de emisión. El código a continuación realiza estas operaciones.

```
In []: exci_nilered = loadtxt('excitacion_NILE_RED.txt')
# Carga el fichero con el espectro de excitacion del fluoróforo en la variable exci_nilered
       long_onda_exc = exci_nilered[:,0] # primera columna
       intens_exc = exci_nilered[:,1] # segunda columna
       emi_nilered = loadtxt('emision_NILE_RED.txt')
       # Carga el fichero con el espectro de emision del fluoróforo en la variable emi_nilered
       long_onda_emi = emi_nilered[:,0] # primera columna
       intens_emi = emi_nilered[:,1] # segunda columna
```

Representamos ambos espectros

```
In []: import mpld3     # el módulo `mpld3` permite mostrar las figuras con la opción de hacer zoom en ellas
       from mpld3 import plugins
       mpld3.enable_notebook()

       fig=figure(figsize=(7,4))
       fill_between(long_onda_exc,intens_exc,0,alpha = 0.2)
       fill_between(long_onda_emi,intens_emi,0,color='r',alpha = 0.2)
       xlabel('Longitud de onda (nm)')
       ylabel('Espectro normalizado')
       text(630,50,'Emission');
       text(500,60,'Excitation');
       plugins.connect(fig, plugins.MousePosition(fontsize=14))
       mpld3.display(fig)
```

Ejercicio 3:

De la figura anterior, indique cuál es el valor del desplazamiento de Stokes y en qué rango de longitudes de onda se produce solapamiento entre los dos espectros. Conteste en esta celda a esta cuestión

---

Cargamos ahora el fichero del espectro de la fuente M530L3 de Throlabs, **M530L3.txt**, que utilizaremos para ilumninar estas partículas y mostramos su espectro superpuesto a los anteriores. El espectro de emisión de la fuente toma valores entre 0 y 1, por tanto para poder visualizarlo en la misma gráfica habrá que multiplicarlo por 100.

```
In []: espect_fuente = loadtxt('M530L3.txt')
       long_onda_fuente = espect_fuente[:,0]
       intens_norm_fuente = 100.0*espect_fuente[:,1]
       mpld3.disable_notebook()
       figsize(7,4)
       plot(long_onda_fuente,intens_norm_fuente,'g')
       fill_between(long_onda_exc,intens_exc,0,alpha = 0.2)
       fill_between(long_onda_emi,intens_emi,0,color='r',alpha = 0.2)
       xlabel(r'$\lambda$ (nm)')
       ylabel('Espectro normalizado')
       axis([400,750,0,100]);
```

Vamos a elegir los filtros para poder medir la señal fluorescente. Para ello tendremos en cuenta los espectros anteriores.

En el laboratorio disponemos de 9 filtros cuyas referencias (*Part number*) son las siguientes:

- Semrock, part number FF01-520/35-25. Fichero **FF01-520_35.txt**
- Semrock, part number FF01-593/40. Fichero **FF01-593_40.txt**
- Semrock, part number FF01-452/45-25. Fichero **FF01-452_45.txt**
- Semrock, part number FF01-550/88-25. Fichero **FF01-550_88.txt**
- Newport, part number 10LF10-633. Fichero **10LF10-633.txt**
- Chroma, part number BLP01-594R-25. Fichero **BLP01-594R.txt**
- Chroma, part number BLP01-514R-25. Fichero **BLP01-514R.txt**
- Semrock, part number Di02-R442, Fichero **Di02-R442.txt**
- Semrock, part number FF562-Di03. Fichero **FF563-di03.txt**
- Semrock, part number 485dclp. Fichero **485dclp.txt**

Ejercicio 4

Mostramos a continuación el enlace a la página web de la empresa Semrock, especializada en la venta de filtros para todo tipo de aplicaciones. Busque en dicha página alguno de estos filtros anotando en esta celda su precio.

Página web de Semrock

Ejercicio 5

Cargue los ficheros espectrales de las transmitancias de estos filtros y representelos. La primera columna corresponde a la longitud de onda en nanómetros y la segunda columna a la transmitancia. Indique dentro de estos, qué filtro de excitación, de emisión y dicroico escogería para las partículas *Nile Red* (Para ello indique el *Part number* que aparece arriba). Argumente la elección realizada. Conteste a esta cuestión en esta celda.

Filtro de excitación:

Filtro de emisión:

Filtro dicroico:

Vamos a comprobar el efecto de los filtros escogidos sobre los espectros cargados anteriormente. En la siguiente celda de código escriba los nombres de los ficheros escogidos en el ejercicio 5 como filtros de excitación, emisión y dicroico. El siguiente código muestra los datos del archivo de transmitancia vs longitud de onda de cada filtro seleccionado junto con los espectros de la fuente, y los de excitación y emisión de las partículas.

Introduzca en la siguiente celda el código los nombres de los ficheros escogidos como filtros de excitación, emisión y dicroico.

```
In []: filtro_exc = loadtxt('nombre_fichero_filtro_excitacion_elegido')
       long_onda_fexc = filtro_exc[:,0]
       espec_exc = 100*filtro_exc[:,1]

       filtro_emi = loadtxt('nombre_fichero_filtro_emision_elegido')
       long_onda_femi = filtro_emi[:,0]
       espec_emi = 100.0*filtro_emi[:,1]

       filtro_dic = loadtxt('nombre_fichero_filtro_dicroico_elegido.txt')
       long_onda_fdic = filtro_dic[:,0]
       espec_dic = 100.0*filtro_dic[:,1]

       fig = figure()
       figsize(7,4)
       plot(long_onda_fuente,intens_norm_fuente,'g')
       fill_between(long_onda_exc,intens_exc,0,color='g',alpha = 0.2)
       fill_between(long_onda_emi,intens_emi,0,color='r',alpha = 0.2)
       xlabel(r'$\lambda$ (nm)')
       ylabel('Espectro normalizado')
       text(630,50,'Emission');
       text(500,60,'Excitation');
       title('NILE RED')
       plot(long_onda_fexc,espec_exc,'b')
       plot(long_onda_femi,espec_emi,'r')
       plot(long_onda_fdic,espec_dic,'black')
       xlabel(r'$\lambda$ (nm)')
       ylabel('Espectro normalizado')
       axis([400,750,0,100]);
```

1.3 Proflavina

Ejercicio 6

- Cargar el fichero del espectro de excitación **excitacion_PROFLAVINA.txt** y el de emisión **emision_PROFLAVINA.txt** de la *proflavina*.

- Representarlo junto con el espectro de emisión de la fuente que se empleará para excitarla, **M455L3.txt**.
- Elegir cuál de los filtros de los que disponemos es el mas adecuado para ver la fluorescencia de la proflavina.
- Representar las transmitancias de los filtros elegidos junto con los espectros de emisión y excitación de la proflavina y el de emisión de la fuente.

Ejercicio 7

Indique cuál es el valor del desplazamiento de Stokes en el caso de la proflavina y en qué rango de longitudes de onda se produce solapamiento entre los dos espectros. Justifique la elección de los filtros. Conteste en esta celda a esta cuestión

1.4 Protoporfirina IX

Ejercicio 8

- Cargar el fichero del espectro de excitación **excitacion_PROTOPORFIRINA.txt** y de emisión **emision_PROTOPORFIRINA.txt** de la *protoporfirina*.

- Representarlo junto con el espectro de emisión de la fuente de iluminación que se empleará para excitarla, **LZ1-10UA00.txt**.
- Elegir cuál de los filtros de los que disponemos es el mas adecuado (si lo hubiera o si fuese necesario) para ver la fluorescencia de la protoporfirina. Representar las transmitancias de los filtros elegidos junto con los espectros emisión y excitación de la protoporfirina y el de emisión de la fuente.

Ejercicio 9

Indique cuál es el valor del desplazamiento de Stokes en el caso de la protoporfirina y en qué rango de longitudes de onda se produce solapamiento entre los dos espectros. Justifique la elección de los filtros. Conteste en esta celda a esta cuestión

Tareas Práctica 4 Nile Red

Sonia Melle Hernández, Oscar Gómez Calderón

1 Óptica Biomédica. Práctica 4

Modificar esta celda incluyendo la información solicitada

NOMBRE Y APELLIDOS:

FECHA DE ENTREGA:

Ejecutar la siguiente celda para cargar el módulo de cálculo científico (%pylab inline).

```
In []: %pylab inline
       import mpld3
       from mpld3 import plugins
```

NOTA: Las dos celdas siguientes cambian las comas decimales por puntos en los ficheros medidos en el laboratorio

```
In []: # ESCRIBIR EN ESTA CELDA LOS NOMBRES DE LOS FICHEROS DEL LED
       # Y  DE LOS FILTROS DICROICO, DE EXCITACIÓN Y DE EMISIÓN
       ## Para cada filenamedicroico = ''
       filenameled = 'nombre1.txt'
       filenamedicroico = 'nombre2.txt'
       filenameemision = 'nombre3.txt'
       filenameexcitacion = 'nombre4.txt'
```

```
In []: # NO TOCAR ESTA CELDA, SOLO EJECUTAR

       with open(filenameled) as f:
           file_str = f.read()
           out = file_str.replace(',', '.')
       with open(filenameled, "w") as f:
           f.write(out)
       ##################
       with open(filenamedicroico) as f:
           file_str = f.read()
           out = file_str.replace(',', '.')
       with open(filenamedicroico, "w") as f:
           f.write(out)
       ##################
       with open(filenameemision) as f:
           file_str = f.read()
           out = file_str.replace(',', '.')
       with open(filenameemision, "w") as f:
           f.write(out)

       ##################
```

```
with open(filenameexcitacion) as f:
    file_str = f.read()
    out = file_str.replace(',', '.')
with open(filenameexcitacion, "w") as f:
    f.write(out)
```

Parte 1: Caracterización de los filtros empleados en el microscopio de fluorescencia para las partículas NILE RED

Pregunta 1:

Representar en la misma gráfica:

- los espectros de transmisión medidos por el alumno para los filtros empleados en el microscopio de las partículas NILE RED.
- el espectro de emisión medido por el alumno para la fuente LED empleada en el microscopio de las partículas NILE RED.
- los espectros de excitación y emisión de las partículas suministrados por el fabricante (emision_NILE_RED.txt, excitacion_NILE_RED.txt)

Pregunta 2 *(Conteste editando esta celda)*:

1. Comente el efecto de cada filtro.
2. ¿Seleccionan las longitudes de onda deseadas en cada caso?
3. ¿Por qué es necesario el filtro de emisión, si disponemos ya del filtro dicroico?
4. ¿Qué efecto esperaría encontrar en la imagen de las partículas si escogemos el filtro de excitación/emisión en longitudes de onda distintas de las que corresponden al máximo de excitación/emisión de las partículas Nile Red?

Parte 2. Tamaño de las partículas fluorescentes

En la siguiente celda vamos a cargar el fichero con la imagen de la red de Ronchi. Como vamos a utilizar esta imagen para calibrar el tamaño de las partículas Nile red, ambas imágenes (la de la red de Ronchi y la de las partículas) deben tener el mismo aumento (10X o 40X). Mostramos la imagen en una figura que nos da en los ejes los píxeles. Las imágenes salvadas tienen un tamaño de 480 píxeles en la dirección vertical y 640 píxeles en la dirección horizontal.

Para cargar la imagen utilizamos el comando *imread*. Para mostrar la figura con la imagen se utiliza el comando *imshow*.

In []:
```
# MODIFICAR SOLO EL NOMBRE DEL FICHERO QUE CONTIENE LA IMAGEN Y EJECUTAR
from scipy.ndimage import imread
iRonchi = imread('Ronchi10X.jpg')
fig = figure()
imshow(iRonchi)
plugins.connect(fig, plugins.MousePosition(fontsize=14))
mpld3.display(fig)
```

Pregunta 3

Estimar a cuantas micras equivale 1 pixel de la imagen. *(Conteste editando esta celda explicando en detalle el procedimiento seguido)*

Para ello, vamos a representar el perfil de intensidad de la imagen de la red de Ronchi y representarlo en función de los píxeles. Sobre dicho perfil vamos a medir con el cursor los píxeles que equivalen a ese número de líneas de la red. A continuación deduciremos el tamaño del pixel en micras sabiendo que la red de Ronchi tiene 40 líneas/mm. Utilizar el código de las dos celdas siguientes.

```
In []: # MODIFICAR EL NOMBRE DEL FICHERO QUE CONTIENE LA IMAGEN
       # ELEGIR LAS COORDENADAS DE LOS PUNTOS INICIAL Y FINAL QUE DETERMINAR EL PERFIL. LUEGO EJECUTAR

       import scipy.ndimage     #importa el modulo scipy.ndimage
       iRonchi = imread('Ronchi40X.jpg',flatten=True)

       #vamos a generar una línea (dando las corredenadas de dos puntos) en la que
       # queremos calcular el perfil de intensidad

       coor_hori0, coor_vert0 = 100, 300 # escribir las coordenadas en pixeles del punto inicial
       coor_hori1, coor_vert1 = 600, 200  # escribir las coordenadas en pixeles del punto final


       #el número de puntos de esa línea lo escribimos en pixeles
       num=round(sqrt((coor_vert0-coor_vert1)**2+(coor_hori0-coor_hori1)**2))
       x, y = linspace(coor_vert0, coor_vert1, num), linspace(coor_hori0, coor_hori1, num)
       # extraemos los valores de la línea empleando una interpolación cúbica
       zi = scipy.ndimage.map_coordinates(iRonchi, vstack((x,y)))
       #pintamos la línea sobre la imagen
       fig = figure(figsize(7,7))
       figure(figsize(7,7))
       imshow(iRonchi)
       plot([coor_hori0, coor_hori1],[coor_vert0, coor_vert1], 'ro-')

       plugins.connect(fig, plugins.MousePosition(fontsize=14))
       mpld3.display(fig)
```

```
In []: #NO TOCAR. SOLO EJECUTAR

       fig=figure(figsize=(10,5))
       plot(zi);xlabel('pixeles')
       plugins.connect(fig, plugins.MousePosition(fontsize=14))
       mpld3.display(fig)
```

Una vez hecha la calibración vamos a determinar el tamaño de las partículas fluorescentes *Nile Red*. Para ello primero vamos a cargar en una variable la imagen de las partículas.

```
In []: mpld3.disable_notebook()

       from scipy.ndimage import imread
       ipart = imread('particulas40X.jpg',flatten=True)
       fig = figure()
       imshow(ipart,cmap='hot')
       plugins.connect(fig, plugins.MousePosition(fontsize=14))
       mpld3.display(fig)
```

Pregunta 4 *(Conteste editando esta celda explicando en detalle el procedimiento seguido)*:

Determinar el tamaño de las partículas fluorescentes *Nile Red*.

De nuevo realizar el perfil de intensidad de una línea de la imagen que donde se encuentra una partícula. Representar este perfil de intensidad en una gráfica, y mover el cursor sobre esa gráfica para obtener el diámetro de la partícula (dé el valor de la anchura del perfil obtenido a media altura en píxeles). Utilizar

el código de las siguientes celdas en donde hay que introducir el nombre del fichero de las partículas y los valores de las coordenadas inicial y final de los puntos de la recta en píxeles. Para obtener el valor de la anchura real (en micras) utilizar la calibración de la pregunta anterior.

In []:
```
# MODIFICAR EL NOMBRE DEL FICHERO QUE CONTIENE LA IMAGEN DE LAS PARTÍCULAS
# ELEGIR LAS COORDENADAS DE LOS PUNTOS INICIAL Y FINAL QUE DETERMINAR EL PERFIL. LUEGO EJECUTAR

import scipy.ndimage     #importa el modulo scipy.ndimage
ipart = imread('particulas40X.jpg',flatten=True)

#vamos a generar una línea (dando las corredenadas de dos puntos) en la que se encuentre una partícula

coor_hori0, coor_vert0 = 169, 225 # escribir las coordenadas en pixeles del punto inicial
coor_hori1, coor_vert1 = 169, 275  # escribir las coordenadas en pixeles del punto final


#el número de puntos de esa línea lo escribimos en pixeles
num=round(sqrt((coor_vert0-coor_vert1)**2+(coor_hori0-coor_hori1)**2))
x, y = linspace(coor_vert0, coor_vert1, num), linspace(coor_hori0, coor_hori1, num)
# extraemos los valores de la línea empleando una interpolación cúbica
zi = scipy.ndimage.map_coordinates(ipart, vstack((x,y)))
#pintamos la línea sobre la imagen
fig = figure(figsize(7,7))
imshow(ipart)
plot([coor_hori0, coor_hori1],[coor_vert0, coor_vert1], 'ro-')
plugins.connect(fig, plugins.MousePosition(fontsize=14))
mpld3.display(fig)
```

In []:
```
#NO TOCAR. SOLO EJECUTAR
# Muestra el corte realizado y que se utilizará para calcular el tamaño de la partícula
fig=figure(figsize=(10,5))
plot(zi);xlabel('pixeles')
plugins.connect(fig, plugins.MousePosition(fontsize=14))
mpld3.display(fig)
```

Introducciónn Teórica Práctica 5

Eduardo Cabrera Granado

1 Teoría Difraccional de la Imagen

Objetivos

1. Manejarse con los conceptos de Función de Dispersión de Punto (PSF) y Función de Transferencia de Modulación (MTF).
2. Describir las aberraciones de onda a través de polinomios de Zernike y ver su efecto en la formación de la imagen.

Introducción

La Óptica Geométrica (OG) proporciona herramientas muy útiles para analizar los sistemas ópticos. Sin embargo, el modelo de propagación de la luz por medio de rayos no incluye los efectos debidos al carácter ondulatorio de la luz. En particular, la OG predice para un sistema estigmático que la imagen de un punto O localizado en el plano objeto es otro punto O' situado en el plano imagen. Esta imagen no se corresponde exactamente con la realidad. En sistemas ópticos perfectos (libres de aberraciones), se encuentra que la imagen de un punto es una mancha extensa, debido a la difracción que sufre la onda luminosa. Este efecto sucede siempre que la luz atraviesa un sistema óptico que obstruye parcialmente los frentes de onda debido a la presencia de diafragmas o por el propio tamaño finito de las lentes.

El ejemplo más sencillo de difracción es el paso de la luz a través de una apertura circular. En este caso, se puede apreciar en una pantalla situada detrás de la lente una mancha circular compuesta por diferentes anillos, y denominada mancha de Airy. El ángulo subtendido por el primer mínimo de irradiancia es $\theta = 1.22\lambda/D$, donde D es el diámetero de la apertura y λ es la longitud de onda de la luz. Debido al uso generalizado de lentes y diafragmas circulares, esta figura aparecerá como imagen O' de un punto objeto O en un sistema libre de aberraciones, alejándose por tanto de las predicciones de la OG y limitando la resolución de dos puntos objeto muy cercanos entre sí.

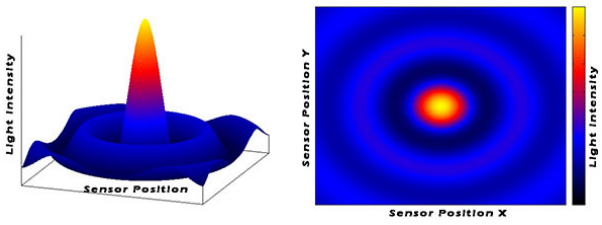


Figure 1: Mancha de Airy

Este caso representativo permite mostrar que la difracción va a jugar un papel importante en la formación de la imagen. Cuando tenemos en cuenta estos efectos en el cálculo de la imagen dada por un sistema óptico hablamos de teoría difraccional de la imagen.

Al analizar la formación de la imagen de objetos extensos, se pueden seguir dos enfoques alternativos:

1. La imagen del objeto extenso está formada por la composición de las imágenes de cada uno de los puntos que forman el objeto. En este enfoque, el elemento fundamental que nos permite obtener la imagen global dada por el sistema es la imagen de un único punto objeto. A esta imagen se le denomina Función de Dispersión de Punto (PSF por sus siglas en inglés) Sabiendo esta información, podemos reconstruir toda la imagen.
2. El objeto extenso puede descomponerse en tramas o redes sinusoidales de distinto periodo y distinta orientación. La imagen resulta de la composición de todas las imágenes correspondientes a cada una de estas tramas. O dicho de otro modo, se analiza la respuesta del sistema en el espacio de frecuencias. Este enfoque nos llevará a definir la Función de Transferencia Óptica (OTF por sus siglas en inglés), la Función de Transferencia de Modulación (MTF) y la Función de Transferencia de Fase (PTF).

Función de Dispersión de Punto (PSF)

La Función de Dispersión de Punto o Point Spread Function (PSF) de un sistema óptico proporciona la imagen de un objeto puntual dada por dicho sistema. Sabiendo qué imagen obtenemos de un único punto, podemos deducir la imagen global de un objeto extenso. La deducción de la irradiancia en la imagen que obtenemos al considerar un objeto puntual no es complicada y se puede consultar en libros más especializados como por ejemplo Introduction to Fourier Optics de Joseph W. Goodman.

El resultado principal es que la Función de Dispersión de Punto es proporcional al cuadrado de la transformada de Fourier de la función pupila de nuestro sistema óptico. Es decir,

$$PSF(x,y) \propto |FT[P(x,y)]|^2$$

donde la función púpila, para el caso de una única lente toma el valor 1 dentro de la lente, y 0 fuera de ella. Esta operación nos da para este caso que la PSF es la función de Airy, la cual ya es conocida dentro del marco de la difracción producida por una abertura circular.

Una consecuencia de lo mostrado anteriormente es que la difracción impone un límite de resolución último para los sistemas ópticos. Incluso en un sistema completamente libre de aberraciones, la imagen de un punto no es un punto, sino una mancha más o menos extensa, dependiendo del diámetro de la lente. Cuando las imágenes de dos puntos muy cercanos se solapen, no podrán distinguirse, existiendo un ángulo mínimo de resolución para discriminar dos puntos objeto puntuales. El criterio más usado para decidir si dos puntos próximos son resolubles por un sistema óptico es el denominado criterio de Rayleigh.

Si en vez de un objeto puntual tenemos un objeto extenso, cada punto del objeto dará lugar en la imagen a una mancha de difracción. Lo podemos interpretar como si cada punto de la imagen dada por la Óptica Geométrica se sustituyera por la PSF del sistema, resultando por tanto en un detrimento de la calidad de la imagen.

Si el objeto lo representamos por su distribución de irradiancia O(x,y), la irradiancia en un punto del plano imagen I(u,v) se obtendrá sumando la contribución de todas las imágenes de cada uno de los puntos del objeto. Al considerar iluminación incoherente, esta suma se realiza sobre las irradiancias resultantes de la imagen de cada punto del plano de observación.

$$I(u,v) = O(x_0,y_0)PSF(u-x_0,v-y_0) + O(x_1,y_1)PSF(u-x_1,v-y_1) + \ldots$$

o lo que es lo mismo, y dado que la distribución de irradiancia es una función contínua,

$$I(u,v) = \int\int O(x,y)PSF(u-x, v-y)dxdy$$

a esta operación se le denomina convolución. Por tanto, dada la PSF de un sistema óptico, podemos hallar la imagen de cualquier objeto extenso.

FIGURE 10.13 Convolution illustrated. The observed image (*left*) is the convolution of the true object (*center*) with the PSF (*right*).

Figure 2: Convolución con PSF

Función de Transferencia de Modulación (MTF)

Podemos caracterizar un sistema óptico no solo por la imagen dada de un objeto puntual, sino analizando cómo se transmiten las diferentes frecuencias espaciales a través del sistema. Si conocemos esta información, y calculamos la descomposición en frecuencias del objeto extenso O(x,y) (mediante su transformada de Fourier), podemos obtener también su imagen. Para entender este enfoque, veamos qué ocurre con la imagen dada por un sistema óptico de un objeto sinusoidal, cuya irradiancia se representa de la siguiente forma,

$$O(x,y) = 1 + m\cos(2\pi(ax+by))$$

donde $m \leq 1$. Este factor representa la amplitud de la modulación, o lo que es lo mismo, el contraste del objeto, definido por,

$$C_{objeto} = \frac{O_{max} - O_{min}}{O_{max} + O_{min}}$$

Por otro lado, a y b son las frecuencias espaciales en las direcciones del eje X y eje Y respectivamente. La frecuencia total será $\sqrt{a^2+b^2}$. La imagen predicha por la OG de este objeto consistirá en la misma función afectada únicamente por el factor de aumento del sistema. Si consideramos los efectos de difracción, la imagen vendrá dada por la convolución de la PSF del sistema con la función O(x,y),

$$I(u,v) = \int\int dxdy[1 + \frac{C_{objeto}}{2}\exp(i2\pi(ax+by)) +$$
$$+\frac{C_{objeto}}{2}\exp(-i2\pi(ax+by))]PSF(u-x, v-y)$$

Si realizamos el cambio de variable $\chi = u - x, d\chi = -dx$ y $\psi = v - y, d\psi = -dy$, considerando que la PSF se encuentra normalizada a área unidad y reagrupando algunos términos, obtenemos finalmente la expresión,

$$I(u,v) = 1 + \frac{C_{objeto}}{2}\exp(i2\pi(au+bv))\int\int d\chi d\psi PSF(\chi,\psi)\exp(-i2\pi(a\chi+b\psi)) + \ldots +$$

$$+\frac{C_{objeto}}{2}\exp(-i(2\pi(au+bv))\int\int d\chi d\psi PSF(\chi,\psi)exp(i2\pi(a\chi+b\psi))$$

En la primera parte de esta expresión, podemos fijarnos que tenemos la transformada de Fourier de la PSF del sistema. A esta función se le denomina Función de Transferencia Óptica (OTF). En nuestro caso, dicha función está evaluada en las frecuencias espaciales a y b. La segunda parte de la expresión muestra el complejo conjugado de la OTF, ya que esta función es una función compleja. Si la escribimos separando su módulo y su fase

$$OTF = \mid OTF \mid \exp(i\Phi_{OTF})$$

la ecuación anterior queda,

$$I(u,v) = 1 + \frac{C_{objeto}}{2} \exp(i2\pi(au+bv)) \mid OTF \mid \exp(i\Phi_{OTF}) +$$
$$+ \frac{C_{objeto}}{2} \exp(-i2\pi(au+bv)) \mid OTF \mid \exp(-i\Phi_{OTF})$$

La cual se puede reescribir finalmente como,

$$I(u,v) = 1 + C_{objeto} \mid OTF(a,b) \mid \cos(2\pi(au+bv) + \Phi_{OTF})$$

El resultado anterior nos muestra que la imagen de un objeto cosenoidal vuelve a ser una función coseno. El módulo de la OTF modifica el contraste de la imagen, que pasa a ser $C_{imagen} = C_{objeto} \mid OTF \mid$ mientras que la fase de la OTF produce un corrimiento de la fase del objeto. Al módulo de la OTF se le denomina Función de Transferencia de Modulación (MTF), y nos da información de cómo el sistema óptico modifica el contraste de la imagen en función de la frecuencia espacial. Por otro lado, a la fase se le denomina Función de Transferencia de Fase (PTF), y nos da cómo se modifica la fase del objeto, es decir, cómo se desplaza la imágen con respecto al objeto.

En la figura a continuación se muestra el efecto de dos MTF distintas sobre la calidad de la imagen. Hay que tener en cuenta que la OTF y la PSF son dos formas distintas de describir un mismo proceso, la formación de la imagen por medio de un sistema óptico. Una PSF más ancha llevará consigo una MTF más estrecha y por tanto una frecuencia de corte menor. Es decir, el sistema será incapaz de reproducir las frecuencias más altas, asociadas a los detalles de la imagen.

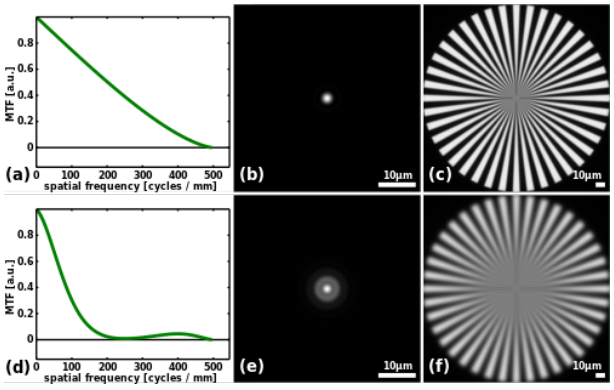


Figure 3: MTF

Aberraciones y polinomios de Zernike

En el anterior desarrollo se ha asumido que si el objeto O es un punto ideal, el frente de ondas emergente de la pupila de salida del sistema optico es un frente de ondas esferico que converge en el punto imagen

O' predicho por la OG. En este caso se dice que el sistema esta únicamente limitado por la difraccion. Sin embargo, la presencia de aberraciones puede modificar enormemente esta descripcion, y por tanto la PSF y la OTF del sistema.

Cabe preguntarse pues como podemos tratar las aberraciones dentro de la teoria difraccional de la imagen. Es decir, como analizar las aberraciones incluyendo la naturaleza ondulatoria de la luz. Si, como hemos comentado, para un sistema perfecto en presencia de un objeto puntual el frente de ondas emergente es una onda esférica, en presencia de aberraciones el frente de ondas se alejara de esta forma ideal. De este modo, podemos caracterizar las aberraciones del sistema por medio de una función W(x,y) que nos dé la separacion en todo punto de la pupila de salida entre el frente de ondas real y el frente de ondas ideal esférico. Concretamente, W(x,y) mide la diferencia en camino óptico entre un rayo que pasa por un punto (x,y) de la pupila de salida del sistema y el rayo paraxial que pasa por ese mismo punto de la pupila de salida.

Esta diferencia de camino optico se traducira en una diferencia de fase entre el frente de ondas real y el ideal $\phi = ikW(x,y)$. Podemos por tanto definir una función pupila generalizada que incluya el tamaño y caracteristicas de la pupila de salida, así como esta fase adicional,

$$P_{Gen}(x,y) = P(x,y)\exp\left(ikW(x,y)\right)$$

De este modo todas las expresiones deducidas anteriormente siguen siendo validas sustituyendo la función pupila P(x,y) por la función pupila generalizada P_{Gen}(x,y), la cual es una magnitud compleja.

Una vez visto cómo introducir las aberraciones en la teoría difraccional de la imagen tenemos que describir la función de aberración W(x,y) así como introducir diferentes magnitudes que nos permitan medir la calidad de la imagen formada por un sistema. Existen diferentes métricas para cuantificar la calidad de imagen de un sistema óptico. Quizás la más sencilla sea aquella que mide en múltiplos de la longitud de onda la mayor separación entre el frente de ondas aberrado y el ideal. Sin embargo, aunque simple, esta métrica resulta de limitada utilidad al dar poca información del frente de ondas real. Por ejemplo un pequeño defecto puede dar lugar a un valor alto de esta separacion, mientras que su efecto sobre la imagen será muy limitado. En la presente práctica vamos a explorar dos métricas ampliamente utilizadas, la dispersión cuadratica media (RMS) del frente de onda y la razón de Strehl.

RMS

La dispersión cuadratica media σ da una medida global de la aberración de onda. En efecto, su expresión es la siguiente,

$$\sigma^2 = < W^2(\rho,\theta) - < W(\rho,\theta) >^2 > = \frac{1}{A}\int\int |W(\rho,\theta) - < W(\rho,\theta) >^2 |d^2\rho$$

Como vemos, utiliza los valores de la función aberración en todo punto. Esta magnitud nos da una medida estadítica de la distancia al frente de ondas ideal.

Razón de Strehl

La razón de Strehl proporciona una métrica de la calidad de la imagen basada en la comparación entre la imagen obtenida por el sistema real y la que se obtendría teóricamente en ausencia de aberraciones. Se define por el cociente entre el máximo de la PSF del sistema con aberraciones y el máximo de la PSF del mismo sistema pero en ausencia de aberraciones. Es decir,

$$Strehl = \frac{max(PSF_{real})}{max(PSF_{ideal})}$$

En el caso en que las aberraciones del sistema no degraden mucho la imagen, se puede demostrar que ambas magnitudes están relacionadas por la expresion,

$$Strehl = \exp[-(2\pi RMS/\lambda)^2]$$

Polinomios de Zernike

La descripcion de la función W(x,y) se realiza normalmente mediante una expansion en serie de polinomios, cada uno de los cuales se atribuye a un tipo de aberración. Debido a la simetría circular de la mayoria de los sistemas ópticos usualmente estos polinomios se expresan en coordenadas polares (ρ, θ) . Además, y con vistas a caracterizar la calidad óptica de la imagen, resulta conveniente utilizar una base de polinomios en la cual el cálculo de las métricas anteriormente vistas fuera sencillo. Estas condiciones se cumplen para los polinomios de Zernike.

Aunque la expresión completa puede parecer difícil (vease por ejemplo en esta página), su forma final no lo es. En la Tabla que se presenta a continuación se puede ver que expresiones toman, mientras que la figura mostrada en esta sección nos permite ver las superficies que definen.

j	n	m	Expresión Zernike	Tipo de aberración
0	0	0	1	Término constante
1	1	-1	$2\ \rho \sin(\theta)$	Tilt
2	1	1	$2\ \rho \cos(\theta)$	Tilt
3	2	-2	$\sqrt{6}\rho^2 \sin(2\theta)$	Astigmatismo
4	2	0	$\sqrt{3}(2\rho^2 - 1)$	Desenfoque
5	2	2	$\sqrt{6}\rho^2 \cos(2\theta)$	Astigmatismo
6	3	-3	$\sqrt{8}(3\rho^3 \sin(3\theta)$	Trefoil
7	3	-1	$\sqrt{8}(3\rho^3 - 2\rho) \sin(\theta)$	Coma
8	3	1	$\sqrt{8}(3\rho^3 - 2\rho) \cos(\theta)$	Coma
9	3	3	$\sqrt{8}(3\rho^3 \cos(3\theta)$	Trefoil
10	4	-4	$\sqrt{10}\rho^4 \sin(4\theta)$	Quadrafoil
11	4	-2	$\sqrt{10}(4\rho^4 - 3\rho^2) \sin(2\theta)$	Astigmatismo secundario
12	4	0	$\sqrt{5}(6\rho^4 - 6\rho^2 + 1)$	Esférica
13	4	2	$\sqrt{10}(4\rho^4 - 3\rho^2) \cos(2\theta)$	Astigmatismo secundario
14	4	4	$\sqrt{10}\rho^4 \cos(4\theta)$	Quadrafoil

Aparte de con los valores de n y m, los polinomios de Zernike se pueden denotar únicamente con un subíndice j. La relación entre j y el par (n,m) viene dada por la siguiente expresión,

$$j = \frac{n(n+2)+m}{2}$$

En la presente práctica utilizaremos la notación de un único subíndice.

Estos polinomios forman una base completa ortogonal en el círculo unidad. Esto quiere decir que cualquier función en el plano puede ser desarrollada como una suma de polinomios de Zernike. Además, su media

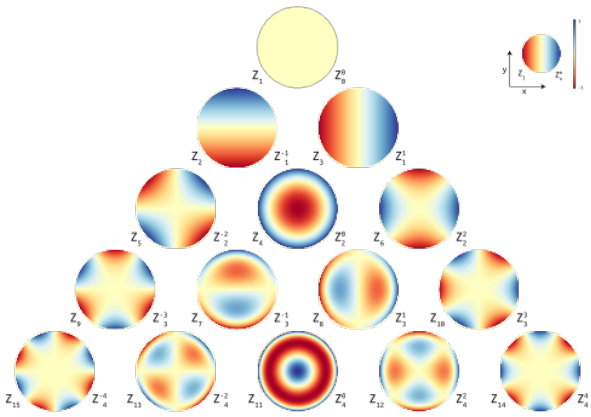


Figure 4: Polinomios de Zernike

es nula y su varianza (σ^2) es igual a 1. Estas dos últimas propiedades tienen como consecuencia que el cálculo de la RMS resulte muy sencillo con estos polinomios. En efecto se puede demostrar que si la función aberración $W(\rho, \theta)$ tiene un desarrollo del tipo,

$$W(\rho,\theta) = c_0 Z_0 + c_1 Z_1 + c_2 Z_2 + \ldots = \sum_i c_i Z_i$$

el cálculo de la RMS vendrá dado únicamente por la expresión, $RMS = \sqrt{\sum_i c_i^2}$.

En el desarrollo de $W(\rho, \theta)$, los coeficientes c_i tienen unidades de longitud y se denominan coeficientes de aberración.

Para saber más:

Página de Wofram Mathworld sobre las propiedades de los Polinomios de Zernike

Trabajo Previo Práctica 5

Eduardo Cabrera-Granado, Oscar Gómez Calderón, Fernando Carreño Sánchez

1 Óptica Biomédica. Trabajo Previo Práctica 5.

NOMBRE Y APELLIDOS:

Modificar esta celda incluyendo la información solicitada

En este Trabajo Previo vamos a estudiar la representación de las aberraciones de onda para luego estudiar en las Tareas cómo afectan a la imagen dada por un sistema óptico.

Para una introducción de los distintos conceptos utilizados en la práctica, así como un resumen de los polinomios de Zernike que se utilizan en este trabajo previo a la realización de la práctica se recomienda leer el documento de Introducción Teórica en formato notebook. También se puede consultar la presentación en PDF sobre nociones básicas de la Teoría Difraccional de la Imagen

1.1 Aberración de onda

Para analizar la formación de la imagen incluyendo los efectos de la naturaleza ondulatoria de la luz (difracción), y dado que la presencia de aberraciones influye enormemente en la imagen final dada por un sistema óptico, tenemos que plantearnos cómo incluirlas dentro del marco de la teoría difraccional de la imagen.

Como para un sistema perfecto en presencia de un objeto puntual el frente de ondas emergente es una onda esférica, la presencia de aberraciones implicará que el frente de ondas se alejará de esta forma ideal. De este modo, podemos caracterizar las aberraciones del sistema por medio de una función W(x,y) que nos dé la separacion en todo punto de la pupila de salida entre el frente de ondas real y el frente de ondas ideal esférico. Concretamente, **W(x,y) mide la diferencia en camino óptico entre un rayo que pasa por un punto (x,y) de la pupila de salida del sistema y el rayo paraxial que pasa por ese mismo punto de la pupila de salida** (vease figura). Si W(x,y) es positiva para un cierto punto, el frente de ondas aberrado se adelanta al ideal en ese punto.

Aberración de onda. Imagen extraída de http://www.hindawi.com

Esta diferencia de camino optico se traducira en una diferencia de fase entre el frente de ondas real y el ideal $\phi = ikW(x, y)$. Esta diferencia de fase la introduciremos en el campo de salida del sistema óptico, permitiendo incluir las aberraciones en nuestro marco de estudio de la formación de la imagen teniendo en cuenta la difracción.

Esta introducción de la aberración de onda deja en el aire cómo se describe esta función W(x,y). También deja en el aire cómo caracterizar estas aberraciones. Es decir, ¿podemos definir alguna variable que nos permita medir la *magnitud* de las aberraciones y por tanto estimar la calidad de la imagen resultante?. En la presente práctica vamos a explorar dos métricas ampliamente utilizadas, la dispersión cuadratica media

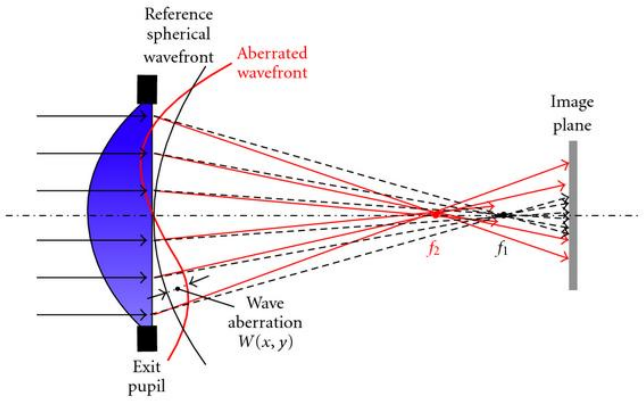


Figure 1: wave aberration

(RMS) del frente de onda (que definiremos en este documento) y la razón de Strehl (a utilizar en el documento de Tareas).

Antes de nada, vamos a cargar los módulos de cálculo científico de Python.

```
In [1]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

Lo primero que necesitamos establecer en el programa son los parámetros físicos de nuestro sistema óptico. El siguiente código fija dichas variables.

Escribe un valor razonable para el diámetro de la púpila

```
In [2]: pupila_diametro= # Diametro de la pupila del sistema en m
        pupila_radio=pupila_diametro/2 # radio de la pupila
```

1.2 Pupila del sistema

A continuación vamos a mostrar la función púpila del sistema sin incluir las aberraciones. Esta función púpila será una función que tome valores de 1 dentro del diámetro definido anteriormente, y 0 fuera de él. Es decir,

$$Pupila(x, y) = \begin{cases} 1 & r \leq r_p \\ 0 & r \geq r_p \end{cases}$$

donde r_p es el radio de la púpila, y $r = \sqrt{x^2 + y^2}$

```
In []: # SOLO EJECUTAR
       fig1 = figure(figsize=(8,6))
       npixels=601 # Numero de pixels (usar un numero impar)

       # Magnitudes adimensionalizadas con el radio de la pupila
       x=linspace(-5,5,npixels) # coordenadas
       dx=x[2]-x[1] # tamaño del pixel (adimensionalziado por el radio de la pupila)
       y=x
```

```
X,Y=meshgrid(x,y); # coordenadas definidas en la matriz
# Funcion pupila (matriz de valores para cada para de coordenadas)
pupila=(X**2+Y**2<1.0)


pcolormesh(x*pupila_radio*1e3,y*pupila_radio*1e3,pupila)
xlabel('mm')
ylabel('mm')
colorbar()
xlim(-pupila_diametro*1e3, pupila_diametro*1e3)
ylim(-pupila_diametro*1e3, pupila_diametro*1e3)

title('PUPILA (sin aberraciones)')
```

1.3 Polinomios de Zernike y aberraciones

A continuación vamos a definir la función aberración de onda W(x,y) la cual se realiza normalmente mediante una expansion en serie de polinomios. Más concretamente, resulta muy conveniente utilizar los denominados polinomios de Zernike para dar la expresión de W(x,y) (Nota: normalmente, y dada la simetría circular de la mayoría de sistemas ópticos, se utilizan coordenadas polares, es decir $W(\rho, \theta)$).

En la Tabla que se presenta a continuación se puede ver que expresiones toman, mientras que la figura mostrada en esta sección nos permite ver las superficies que definen.

j	n	m	Expresión Zernike	Tipo de aberración
0	0	0	1	Término constante
1	1	-1	$2\,\rho\sin(\theta)$	Tilt
2	1	1	$2\,\rho\cos(\theta)$	Tilt
3	2	-2	$\sqrt{6}\rho^2\sin(2\theta)$	Astigmatismo
4	2	0	$\sqrt{3}(2\rho^2-1)$	Desenfoque
5	2	2	$\sqrt{6}\rho^2\cos(2\theta)$	Astigmatismo
6	3	-3	$\sqrt{8}(3\rho^3\sin(3\theta)$	Trefoil
7	3	-1	$\sqrt{8}(3\rho^3-2\rho)\sin(\theta)$	Coma
8	3	1	$\sqrt{8}(3\rho^3-2\rho)\cos(\theta)$	Coma
9	3	3	$\sqrt{8}(3\rho^3\cos(3\theta)$	Trefoil
10	4	-4	$\sqrt{10}\rho^4\sin(4\theta)$	Quadrafoil
11	4	-2	$\sqrt{10}(4\rho^4-3\rho^2)\sin(2\theta)$	Astigmatismo secundario
12	4	0	$\sqrt{5}(6\rho^4-6\rho^2+1)$	Esférica
13	4	2	$\sqrt{10}(4\rho^4-3\rho^2)\cos(2\theta)$	Astigmatismo secundario
14	4	4	$\sqrt{10}\rho^4\cos(4\theta)$	Quadrafoil

Aparte de con los valores de n y m, los polinomios de Zernike se pueden denotar únicamente con un subíndice j. La relación entre j y el par (n,m) viene dada por la siguiente expresión,

$$j = \frac{n(n+2)+m}{2}$$

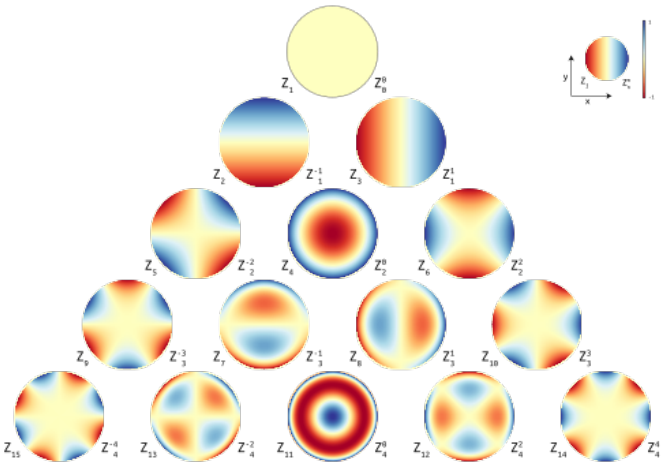


Figure 2: Polinomios de Zernike

En la presente práctica utilizaremos la notación de un único subíndice.

Para ello definimos los distintos polinomios de Zernike. La aberración de onda $W(\rho,\theta)$ la podremos expresar como una suma de estos polinomios cada uno de ellos pesado con un coeficiente distinto. Es decir,

$$W(\rho,\theta) = \sum_i C_i \text{Zernike}(i,\rho,\theta)$$

La suma anterior puede ser, en teoría, infinita. Es decir, podemos definir infinitos polinomios de Zernike con sus correspondientes coeficientes. En la celda de código a continuación se definen los 15 primeros polinomios de Zernike, con los que trabajaremos en la práctica. *Un buen ejercicio puede ser cuestionarse si ese número de polinomios es suficiente para describir las aberraciones de un sistema óptico como el ojo humano, y si hay algún caso en el que necesitemos más.*

In []:
```
# EJECUTAR SIN TOCAR

RHO=sqrt(X**2+Y**2)
THETA=arctan2(Y,X)
############################
# Polinomios de Zernike
############################
zernike = zeros((15,X.shape[0],X.shape[1]))

zernike[0] = ones(X.shape) # n = 0, m= 0. Término constante
zernike[1] = 2.0*RHO*sin(THETA) # n = 1, m = -1. Tilt Y
zernike[2] = 2.0*RHO*cos(THETA)# n = 1, m = 1. Tilt X
#------
zernike[3] = sqrt(6)*(RHO**2)*sin(2*THETA) #Astigmatismo primario Y
zernike[4] = sqrt(3)*(2*RHO**2-1) #Desenfoque
zernike[5] = sqrt(6)*(RHO**2)*cos(2*THETA) #Astigmatismo primario X
#------
zernike[6] = sqrt(8)*(RHO**3)*sin(3*THETA) #Trefoil Y
zernike[7] = sqrt(8)*(3*RHO**3-2*RHO)*sin(THETA) #Coma Y
zernike[8] = sqrt(8)*(3*RHO**3-2*RHO)*cos(THETA) #Coma X
zernike[9] = sqrt(8)*(RHO**3)*cos(3*THETA) #Trefoil X
```

```
#------
zernike[10] = sqrt(10)*(RHO**4)*sin(4*THETA) #Cuadrafoil Y
zernike[11] = sqrt(10)*(4*RHO**4-3*RHO**2)*sin(2*THETA) #Astigmatismo secundario Y
zernike[12] = sqrt(5)*(6*RHO**4-6*RHO**2+1) #Esférica
zernike[13] =sqrt(10)*(4*RHO**4-3*RHO**2)*cos(2*THETA)#Astigmatismo secundario X
zernike[14] = sqrt(10)*(RHO**4)*cos(4*THETA) #Cuadrafoil X

print 'definidos los polinomios de Zernike en coordenadas normalizadas'
```

Ejercicio 1

Ejecutar la siguiente celda de código y en las cajas de texto que aparecen a continuación, seleccionar los índices necesarios para dibujar los polinomios de Zernike correspondientes a las aberraciones de astigmatismo y coma (veáse la tabla de polinomios de Zernike anterior) y asignarles valores que se consideren razonables (en unidades del S.I.). No hace falta salvar las figuras. Editando esta celda, contestar a los puntos que se indican a continuación.

- **Astigmatismo**
  - Índices asociados a la aberración:
  - Valor escogido:
  - Simetrías que presenta el perfil espacial:
  - Zonas del perfil donde el frente de onda se encuentra adelantado con respecto a la esfera de referencia:
  - Zonas del perfil donde el frente de onda se encuentra retrasado con respecto a la esfera de referencia:
- **Coma**
  - Índices asociados a la aberración:
  - Valor escogido:
  - Simetrías que presenta el perfil espacial:
  - Zonas del perfil donde el frente de onda se encuentra adelantado con respecto a la esfera de referencia:
  - Zonas del perfil donde el frente de onda se encuentra retrasado con respecto a la esfera de referencia:

In [5]:
```
# NO TOCAR. SOLO EJECUTAR
# Aparecerán las cajas de texto a continuación
# Una vez cambiado el valor de uno de los campos se carga en el programa
# No hace falta ejecutar de nuevo.
from IPython.html.widgets import FloatTextWidget,interact

wind = FloatTextWidget(description='Indice del Coeficiente',value=0)
wcoef = FloatTextWidget(description='Valor del Coeficiente',value=0.0)

display(wind,wcoef)
```

In []:
```
# NO TOCAR. SOLO EJECUTAR
# Coeficientes inicializados a cero
C=zeros(15) # 14 +1 para que el indice se corresponda al polinomio de zernike
C[wind.value] = wcoef.value
```

```
# Funcion que devuelve la aberración

def W(Coeficientes,zernike):
    # Devuelve la aberracion de onda (Nota: coordenadas normalizadas al radio de la pupila)
    return sum([Coeficientes[i]*zernike[i,:,:] for i in range(len(Coeficientes))],axis=0)

#Representación de la aberración
##################
# Incluimos las coordenadas con dimensiones
Xdim = X*pupila_radio*1e3 # en mm
Ydim = Y*pupila_radio*1e3 # en mm

# Introducimos la pupila del sistema en la aberración
Aberracion = W(C,zernike)*pupila

fig2 = figure(figsize(7,5))
pcolormesh(Xdim,Ydim,Aberracion,cmap='coolwarm') # Representamos
                                                 #la función aberración
                                                 #multiplicada por la pupila
colorbar()
xlim(-pupila_diametro*1e3,pupila_diametro*1e3)
ylim(-pupila_diametro*1e3,pupila_diametro*1e3)
xlabel('X')
ylabel('Y');
```

1.4 RMS

Una métrica muy utilizada para indicar una medida global de la aberración de onda es la dispersión cuadrática media o RMS. El uso de polinomios de Zernike simplifica el cálculo de esta medida pues si $W(\rho,\theta) = \sum_i C_i \text{Zernike}_i(\rho,\theta)$, la RMS se calcula mediante la siguiente expresión,

$$RMS = \sqrt{\sum_i C_i^2}$$

Ejercicio 2

Vamos a calcular la RMS de una aberración dada. Para ello, disponemos de los coeficientes de los polinomios de Zernike de un paciente en el fichero *Zernike_paciente.dat*. El diámetro de su pupila es de 5.4 mm. Se pide:

1. Cargar el fichero de coeficientes (expresados en S.I.)
2. Dibujar la aberración de onda
3. Calcular la RMS del sistema

Notas:

- En la anterior celda se ha definido la función `W` que nos permite dibujar la aberración de onda para unos valores de los coeficientes. Llamar a esta función modificando el argumento de los coeficientes de Zernike que describen la aberración.
- Hay que tener en cuenta que el radio de la pupila es ahora distinto.
- Utilizar el comando `pcolormesh` al igual que en la celda anterior para dibujar la aberración de onda.

1.5 Polinomios de Zernike y refracción ocular

La refracción ocular (esfera, cilindro y eje) puede obtenerse a partir de los valores de los coeficientes de Zernike hasta orden 2, según las siguientes expresiones (en notación de cilindro negativo):

$$S = -\frac{4\sqrt{3}c_4}{R^2} + \frac{2\sqrt{6}\sqrt{c_3^2 + c_5^2}}{R^2}$$

$$C = -\frac{4\sqrt{6}\sqrt{c_3^2 + c_5^2}}{R^2}$$

$$\theta = \frac{1}{2}\arctan(\frac{c_3}{c_5})$$

donde R es el radio de la pupila.

También, por supuesto, podemos hacer el paso inverso. De los datos de esfera, cilindro y eje, podemos calcular los coeficientes de Zernike hasta orden 2. Órdenes superiores tendrían que ser obtenidos por otros medios.

Las expresiones finales que nos dan los coeficientes c_3, c_4 y c_5 a partir de S, C y θ son las siguientes.

$$c_3 = \frac{CR^2}{2\sqrt{6}}\tan(2\theta)\cos(2\theta)$$

$$c_4 = -\frac{SR^2}{4\sqrt{3}} - \frac{CR^2}{8\sqrt{3}}$$

$$c_5 = \frac{1}{\sqrt{6}}\frac{CR^2\text{sen}(2\theta)}{\tan(2\theta)}$$

Ejercicio 3

1. Calcula la refracción ocular (esfera, cilindro y eje) del ejemplo de aberraciones cargado en el **Ejercicio 2** implementando las anteriores expresiones para el cálculo de S, C y θ.
2. Dados los valores de esfera, cilindro y eje que se especifican a continuación, calcular los coeficientes de los polinomios de Zernike c_3, c_4 y c_5 (orden 2).

$S = 0.5\text{D}, C = 0.1\text{D}, \theta = 24^o$

Notas:

- Las funciones tangente, coseno y seno se implementan en Python mediante los comandos: `tan`, `cos` y `sin` respectivamente. Así, sen(0.5) se escribiría `sin(0.5)`. Los ángulos han de introducirse en radianes.
- La función arctan(a/b) se implementa mediante el comando `arctan(a/b)`, devolviendo el resultado en radianes.

Polinomios de Zernike y Refracción Ocular. Efecto de aberraciones de alto orden

Las expresiones anteriores que permiten extraer los valores de esfera, cilindro y eje a partir de los coeficientes de la expansión en polinomios de Zernike de la aberración de onda no consideran contribuciones a estos parámetros más allá de las aberraciones de segundo orden. Sin embargo, esta suposición no es cierta y puede haber contribuciones de las aberraciones de alto orden. Este efecto se produce por la presencia de términos en ρ^2 en los polinomios de Zernike que describen estas aberraciones de alto orden, lo que contribuye a la curvatura en un punto de la superficie que define el frente de ondas. Teniendo en cuenta que la curvatura de un frente de onda determina cómo se focaliza el haz resulta intuitivo comprender que estas aberraciones pueden afectar a los valores de la refracción ocular. Por supuesto, a menor valor de las aberraciones de alto orden, menor será su efecto.

Las expresiones completas incluyendo estos nuevos términos son más complejas que las mostradas anteriormente y por tanto no se pedirá implementarlas en este documento. Sin embargo, a continuación se da una función denominada `Zernike_to_refrac` que permite realizar este cambio de coeficientes de Zernike a esfera, cilindro y eje simplemente dando el vector de coeficientes y el diámetro pupilar. Además, un último argumento denominado *tipo* permite elegir entre el cálculo considerando u omitiendo los coeficientes de alto orden (aberración esférica y astigmatismo secundario)

Ejercicio 4

1. Calcula para el ejemplo de aberraciones cargado en el **Ejercicio 3**, y utilizando la función `Zernike_to_refrac`, los valores de esfera, cilindro y eje sin (tipo=2) y con (tipo=4) aberraciones de alto orden.
2. Compara e interpreta ambos resultados.

Nota: Ejecuta primero la celda en donde se define la función `Zernike_to_refrac` para cargarla en el sistema. A continuación crea una nueva celda para realizar los cálculos solicitados.

```
In [23]: #NO TOCAR. Esta función permite obtener Esfera, cilindro y
         #eje considerando aberraciones de bajo orden únicamente (tipo=2)
         # o bien aberraciones también de alto orden (tipo=4)
         def Zernike_to_refrac(CC,dp,tipo):
             #función que obtiene la esfera, el cilindro y el eje a partir de los valores
             #de los coeficientes de aberración, para una cierta pupila
             #
             #C---->Coeficientes de los polinomios en metros
             #dp--->Tamaño de la pupila en metros
             #tipo->esta variable puede valer 0 ó 1
             #      si vale 0-->calcula la esfera, el cilindro y el eje
             #      empleando solamente los coeficientes de bajo orden
             #      si vale 1-->calcula la esfera, el cilindro y el eje
             #     teniendo en cuenta la esférica y el astigmatismo secundarios
             # Devuelve
             # Esfera, Cilindro y Eje (por ese orden)
             if (tipo==2):
                 caso=0.0
             if (tipo==4):
                 caso=1
             PSE= (-CC[4]*4.0*sqrt(3.0) + caso*CC[12]*12*sqrt(5))/(dp/2.0)**2 #esfera equivalente
             PJ0= (-CC[5]*2.0*sqrt(6.0) + caso*CC[13]*6*sqrt(10))/((dp/2.0)**2) #J_0
             PJ45=(-CC[3]*2.0*sqrt(6.0) + caso*CC[11]*6*sqrt(10))/((dp/2.0)**2 )#J_45
             #---
             alfa=arctan(PJ45/PJ0)*(180.0/pi)*0.5

             #---
```

```
if (alfa==90):
    EJE=90
    CIL=2*PJ0
    ESF=PSE-CIL/2.0
elif (alfa==45):
    EJE=45
    CIL=-2.0*PJ45
    ESF=PSE-CIL/2.0
else:
    EJE=alfa;
    alfaRAD=arctan(PJ45/PJ0)*(1.0/2.0)
    CIL=-2*sqrt(PJ0**2 + PJ45**2)#2*PJ0/cos(2.0*alfaRAD)
    ESF=PSE-CIL/2.0

#----------------
return ESF,CIL,EJE;
#----------------
```

Tareas Práctica5

Eduardo Cabrera-Granado, Oscar Gómez Calderón, Fernando Carreño Sánchez

1 Tareas Práctica 5. Teoría Difraccional de la Imagen

1.1 Pasos previos

Primero cargamos en el documento todas las funciones de cálculo y dibujo que vamos a utilizar para obtener los resultados

In []:
```
%pylab inline
import mpld3
from mpld3 import plugins
mpld3.enable_notebook()
```

A continuación definimos los parámetros físicos del sistema con el que vamos a trabajar en esta parte de la práctica:

In []:
```
pupila_diametro=5.4e-3 # Diametro de la pupila del sistema en m
indice=4.0/3 # índice del medio
zi=22.55e-3 # distancia de la pupila al plano imagen en m
lambda0=500.0e-9 # longitud de onda en el vacio en m

pupila_radio=pupila_diametro/2 # radio de la pupila
lambdam=lambda0/indice # longitud de onda en el medio
```

Pupila del sistema, polinomios de Zernike y Aberración de Onda

Tanto la pupila, como la definición de los polinomios de Zernike y la aberración de onda los necesitaremos en este documento. Al haber sido tratados anteriormente en el Trabajo Previo, únicamente aquí se reproduce su definición para poder usarlos más adelante.

In []:
```
# SOLO EJECUTAR

############
#Pupila
##########
npixels=601 # Numero de pixels (usar un numero impar)
# Magnitudes adimensionalizadas con el radio de la pupila
x=linspace(-5,5,npixels) # coordenadas
dx=x[2]-x[1] # tamaño del pixel (adimensionalziado por el radio de la pupila)
y=x
X,Y=meshgrid(x,y); # coordenadas definidas en la matriz
# Funcion pupila (matriz de valores para cada para de coordenadas)
```

```
pupila=(X**2+Y**2<1.0)

############
# Polinomios de Zernike hasta j = 14
############
RHO=sqrt(X**2+Y**2)
THETA=arctan2(Y,X)

zernike = zeros((15,X.shape[0],X.shape[1]))

zernike[0] = ones(X.shape) # n = 0, m= 0. Término constante
zernike[1] = 2.0*RHO*sin(THETA) # n = 1, m = -1. Tilt Y
zernike[2] = 2.0*RHO*cos(THETA)# n = 1, m = 1. Tilt X
#------
zernike[3] = sqrt(6)*(RHO**2)*sin(2*THETA) #Astigmatismo primario Y
zernike[4] = sqrt(3)*(2*RHO**2-1) #Desenfoque
zernike[5] = sqrt(6)*(RHO**2)*cos(2*THETA) #Astigmatismo primario X
#------
zernike[6] = sqrt(8)*(RHO**3)*sin(3*THETA) #Trefoil Y
zernike[7] = sqrt(8)*(3*RHO**3-2*RHO)*sin(THETA) #Coma Y
zernike[8] = sqrt(8)*(3*RHO**3-2*RHO)*cos(THETA) #Coma X
zernike[9] = sqrt(8)*(RHO**3)*cos(3*THETA) #Trefoil X
#------
zernike[10] = sqrt(10)*(RHO**4)*sin(4*THETA) #Cuadrafoil Y
zernike[11] = sqrt(10)*(4*RHO**4-3*RHO**2)*sin(2*THETA) #Astigmatismo secundario Y
zernike[12] = sqrt(5)*(6*RHO**4-6*RHO**2+1) #Esférica
zernike[13] =sqrt(10)*(4*RHO**4-3*RHO**2)*cos(2*THETA)#Astigmatismo secundario X
zernike[14] = sqrt(10)*(RHO**4)*cos(4*THETA) #Cuadrafoil X

############
# Aberración de onda
############

def W(Coeficientes,zernike,pupila):
    # Devuelve la aberracion de onda (Nota: coordenadas normalizadas al radio de la pupila)
    return pupila*sum([Coeficientes[i]*zernike[i,:,:] for i in range(len(Coeficientes))],axis=0)
```

La definición de los coeficientes del desarrollo en polinomios de Zernike de la aberración de onda la dejamos para los casos que vamos a estudiar.

1.2 Tarea 1. Pupila Generalizada, PSF y cálculo de la imagen final

Pupila Generalizada

Para incluir la aberración de onda en el cálculo de la imagen dada por un sistema óptico, es necesario definir una función pupila generalizada que las incluya. Esto se realiza utilizando la expresión:

$$P_{gen} = P(x,y)e^{ikW(x,y)}$$

siendo $k = \frac{2\pi}{\lambda_m}$ con λ_m la longitud de onda en el medio, $P(x,y)$ la función pupila y por último $W(x,y)$ la función aberración que se expresa como suma de polinomios de Zernike. Esta pupila generalizada ha de redefinirse con la aberración correspondiente si cambiamos los coeficientes de Zernike.

```
In []: # SOLO EJECUTAR
       # Pupila generalizada
       def pupila_gen(Coeficientes,pupila):
            return pupila*exp(1.0j*(2*pi/lambdam)*W(Coeficientes,zernike,pupila))
```

PSF

Ahora calculamos la Función de Dispersión de Punto (PSF), la cual proporciona la imagen de un objeto puntual dada por el sistema óptico. Esta función viene determinada por la pupila del sistema, mediante la expresión,

$$PSF(x,y) \propto |FT[P(x,y)]|^2$$

donde FT significa transformada de Fourier y $P(x,y)$ es la función pupila de nuestro sistema óptico.

Si el objeto en lugar de ser puntual, tiene una extensión espacial, la imagen la obtendremos superponiendo la contribución de todas las imágenes de cada uno de los puntos del objeto. Al considerar iluminación incoherente, esta superposición se realiza sobre las irradiancias resultantes de la imagen de cada punto del plano de observación. O dicho de otro modo, la imagen se obtiene mediante la convolución de la función PSF con el objeto.

$$I(u,v) = \int\int O(x,y)PSF(u-x,v-y)dxdy$$

Si el sistema presenta aberraciones, en la ecuación de definición de la PSF, tendremos que sustituir la función pupila $P(x,y)$ por la función pupila generalizada $P_{gen} = P(x,y)e^{ikW(x,y)}$ vista anteriormente.

```
In []: # SOLO EJECUTAR
       ######################################################################
       # PSF
       ######################################################################

       # Calcula la FT, la centra y su modulo al cuadrado es la PSF
       def calcula_PSF(C,pupila):
            '''
            calcula_PSF(C,pupila)
            función para calcular la PSF del
            sistema óptico. Llama a la función
            pupila_gen(C,pupila) para calcular la pupila generalizada
            C = coeficientes del desarrollo en polinomios de Zernike (en m)
            pupila = matriz que nos da la pupila del sistema
            '''
            tfpupila=fftshift(fft2(pupila_gen(C,pupila)))
            psf_sin_norm = (abs(tfpupila))**2 # Modulo al cuadrado por iluminación incoherente
            return psf_sin_norm/sum(sum(psf_sin_norm)) # PSF normalizada para que la energía total sea 1

       #Representación
       def dibuja_PSF(psf,lambdam,zi,pupila_radio,npixels):
            '''
            dibuja_PSF(psf,lambdam,zi,pupila_radio,npixels)
            función para dibujar la psf previamente
            calculada con la función calcula_PSF
            psf = psf calculada con calcula_PSF
            lambdam = longitud de onda en el medio en m
            zi = distancia de la pupila al plano imagen en m
```

```
    pupila_radio = radio de la pupila en m
    npixels = número de pixels utilizado en la definición de la pupila
    '''
    ##############
    # Define las frecuencias espaciales
    fx=linspace(-0.5,0.5,npixels) # frecuencias adimensionales (en 1/pixels) desde -0.5 a 0.5
    fx=fx/(dx*pupila_radio) # frecuencias con dimensiones (en 1/m)
    fy=fx
    # Nuevas coordenadas espaciales
    xnew=fx*lambdam*zi
    ynew=xnew
    dxnew=xnew[2]-xnew[1] # tamaño del pixel
    [Xnew,Ynew]=meshgrid(xnew,ynew); # coordenadas definidas en la matriz
    # Pinta la PSF
    fig = figure(figsize=(14,6))
    subplot(1,2,1)
    imshow(psf,extent=(xnew[0]*1e6,xnew[-1]*1e6,ynew[0]*1e6,ynew[-1]*1e6),cmap='afmhot')
    xlabel('micras')
    ylabel('micras')
    title('PSF')
    colorbar()

    # Pinta perfil de la PSF
    pixelcentro=(npixels+1)/2; # pixel central
    psfy=psf[:,pixelcentro] # perfil PSF en y
    psfx=psf[pixelcentro,:] # perfil PSF en x

    subplot(1,2,2)
    plot(ynew*1e6,psfy,'k',xnew*1e6,psfx,'r')
    xlabel('micras')
    ylabel('PSF perfil')
    legend(('en y','en x'))
    return 'ok'
```

Imagen Final

Dado un objeto, que nosotros leeremos de un archivo y cargaremos en una variable `objeto`, si sabemos la PSF del sistema podemos calcular la imagen que obtendremos con dicho sistema óptico. El siguiente código carga el archivo deseado en la variable `objeto`, dado el tamaño del objeto y la distancia de observación, calcula la imagen dada por la Óptica Geométrica. Finalmente, calcula la imagen incluyendo las aberraciones. Para ello debemos realizar la convolución de la PSF del sistema con la imagen paraxial que obtendríamos por la Óptica Geómetrica, la cual es simplemente el objeto multiplicado por un factor de escala (el aumento del sistema).

```
In []: # SOLO EJECUTAR
       from scipy.ndimage import imread
       from scipy.interpolate import interp2d
       from scipy.signal import convolve2d

       def calc_imagen(C,objeto,distancia):
           num_pixeles_x = objeto.shape[0]
           num_pixeles_y = objeto.shape[1]
           pixel_tam_x = objeto_tam/num_pixeles_x
           #pixel_tam_y = objeto_tam/num_pixeles_y
           x_obj = arange(num_pixeles_x)*pixel_tam_x*1e3 # en mm
           y_obj = arange(num_pixeles_y)*pixel_tam_x*1e3 # en mm
```

```
# Definimos el angulo subtendido por el objeto en la direccion vertical
# y calculamos el tamagno en el plano imagen de la imagen paraxial.
thetarad=objeto_tam/distancia #angulo en radianes
# angulo refractado
thetatrans=thetarad/indice
# tamagno en vertical (en metros) de la imagen paraxial
imagenparax_tam=zi*tan(thetatrans)
pixel_tam_imagparax = imagenparax_tam/num_pixeles_y
ximagparax = arange(num_pixeles_x)*pixel_tam_imagparax*1e3 # en mm
yimagparax = arange(num_pixeles_y)*pixel_tam_imagparax*1e3 # en mm
# Pinta la imagen paraxial
#ax = fig.add_subplot(121, aspect='equal')
figim = figure(figsize=(14,6))
subplot(1,2,1)
imshow(objeto,extent=(x_obj[0],x_obj[-1],y_obj[0],y_obj[-1]),cmap='gray')
xlabel('mm')
ylabel('mm')
xlim(0,objeto_tam*1e3)
ylim(0,objeto_tam*1e3)
#title('Imagen paraxial',fontsize=16)
title('Objeto',fontsize=16)


######################################################################
# 2. INTERPOLA LA PSF PARA QUE COINCIDA CON EL OBJETO
######################################################################
#Define las frecuencias espaciales
fx=linspace(-0.5,0.5,npixels) # frecuencias adimensionales (en 1/pixels) desde -0.5 a 0.5
fx=fx/(dx*pupila_radio) # frecuencias con dimensiones (en 1/m)
fy=fx
# Nuevas coordenadas espaciales
xnew=fx*lambdam*zi
ynew=xnew
dxnew=xnew[2]-xnew[1] # tamaño del pixel
npixelsi=(max(xnew)-min(xnew))/pixel_tam_imagparax

xnewi=arange(min(xnew),max(xnew),pixel_tam_imagparax)
ynewi=xnewi
[Xnewi,Ynewi]=meshgrid(xnewi,ynewi) # coordenadas definidas en la matriz
funPSFint = interp2d(xnew,ynew,calcula_PSF(C,pupila))
psfi = funPSFint(xnewi,ynewi)
psfi=psfi/(sum(psfi)) # Normalizamos la PSF para que la energia total sea 1

######################################################################
# CALCULA SU IMAGEN CON LA PSF
######################################################################

imagen=convolve2d(objeto,psfi,'same')
#ax1 = fig.add_subplot(122, aspect='equal')
subplot(1,2,2)
imshow(imagen,extent=(ximagparax[0],ximagparax[-1],yimagparax[0],yimagparax[-1]),cmap='gray')
#yaxis.tick_right()
xlabel('mm')
ylabel('mm')

xlim(0,ximagparax[-1])
ylim(0,yimagparax[-1])
title('Imagen con aberraciones',fontsize=16)
```

```
        return imagen
```

Si queremos calcular la PSF del sistema y visualizarla, seguiríamos los siguientes pasos:

1. Definición de la aberración de onda por medio de los coeficientes de los polinomios de Zernike
2. Cálculo de la PSF por medio de la función `psf` (En esta función, se incluye la llamada a `pupila_gen` que nos devuelve la pupila generalizada, incluyendo aberraciones)
3. Si se quiere dibujar, utilizaríamos la función `dibuja_PSF`.

Por ejemplo, podríamos visualizar la PSF del sistema óptico caracterizado por unos coeficientes del desarrollo en polinomios de Zernike de la aberración de onda almacenados en el archivo `ejemplo_Zernike_psf.dat`

```
In []: # SOLO EJECUTAR
       coefs = loadtxt('ejemplo_Zernike_psf.dat')
       print coefs
       psf = calcula_PSF(coefs,pupila)
       dibuja_PSF(psf,lambdam,zi,pupila_radio,npixels)
```

Si queremos calcular la imagen definiendo una nueva aberración de onda seguiríamos los siguientes pasos:

1. Definición de la aberración de onda por medio de los coeficientes de los polinomios de Zernike
2. Cálculo de la imagen final por medio de la función `calc_imagen` (Nota: hay que definir un objeto, su tamaño y una distancia a la que se observaría dicho objeto).

Por ejemplo, utilizando los coeficientes anteriormente cargados,podemos calcular la imagen de la letra E que cargamos del archivo `E1.jpg`, de la siguiente forma.

```
In []: # SOLO EJECUTAR
       objeto = imread('E1.jpg',flatten=True)
       objeto_tam = 0.02 # 2 cm de lado lateral de toda la imagen (la letra E ocupa menos)
       distancia = 1.0 # en m
       imagen=calc_imagen(coefs,objeto,distancia)
```

Ejercicio 1

1. Dibujar la PSF y la imagen predicha para la letra E que se encuentra en el fichero `E1.jpg` para una aberración de onda caracterizada por un valor de S = 0.1 D (esfera), suponiendo un valor nulo para los coeficientes del desarrollo de Zernike no relacionados con S. Utilizar para visualizar la imagen de la letra E una distancia de 1.5 m y un tamaño del objeto de 2 cm.
2. Repetir el apartado anterior para S = 0.3 D. ¿Qué ha provocado en la PSF un aumento de S?
3. Dibujar la PSF y la imagen predicha para la letra E que se encuentra en el fichero `E1.jpg` para una aberración de onda caracterizada por un valor de C = 0.1 D (cilindro), y por un valor de $\theta = 24^o$ (eje), suponiendo un valor nulo para los coeficientes del desarrollo de Zernike no relacionados con C ni con θ.
4. Repetir el apartado anterior para C = 0.3 D y $\theta = 24^o$ (eje). Comentar los cambios en la PSF al aumentar el valor de C.
5. Repetir el apartado 4 para C = 0.1 D y $\theta = 80^o$ (eje).Comentar los cambios en la PSF al cambiar el valor de θ con respecto al estudiado en el apartado 3.

1.3 Tarea 2. Razón de Strehl

La razón de Strehl proporciona una métrica de la calidad de la imagen basada en la comparación entre la imagen obtenida por el sistema real y la que se obtendría teóricamente en ausencia de aberraciones. Se define por el cociente entre el máximo de la PSF del sistema con aberraciones y el máximo de la PSF del mismo sistema pero en ausencia de aberraciones. Es decir,

$$Strehl = \frac{max(PSF_{real})}{max(PSF_{ideal})}$$

En el caso en que las aberraciones del sistema no degraden mucho la imagen, se puede demostrar que ambas magnitudes están relacionadas por la expresion,

$$Strehl = \exp[-(2\pi RMS/\lambda)^2]$$

En este apartado vamos a calcular la razón de Strehl atendiendo a su definición, es decir, como el cociente entre los máximos de las PSF correspodientes al sistema con aberraciones y sin ellas (sistema ideal únicamente limitado por la difracción, en donde tendremos la mancha de Airy como PSF). Posteriormente, comprobaremos si la expresión aproximada anterior es válida en el caso planteado.

Ejercicio 2

1. Definir un array de 14 coeficientes correspondientes al sistema ideal. A continuación calcular y dibujar la PSF de este sistema ideal.
2. Calcular y dibujar PSF del sistema óptico con las aberraciones definidas por el conjunto de coeficientes almacenados en el archivo `Zernike_paciente.dat`.
3. Calcular la Razón de Strehl de este sistema según su definición y según la fórmula aproximada (utilizando el valor de la RMS calculado en el Trabajo Previo). Comentar el resultado

NOTA: Para calcular el máximo de una array 2D (como es la PSF), utilizar `nombre_variable.max()`

1.4 Tarea 3. MTF y Contraste de la imagen

MTF

La Función de Transferencia de Modulación (MTF por sus siglas en inglés) nos da información de cómo varía el contraste en la imagen. Más concretamente, si tenemos un objeto cosenoidal caracterizado por una cierta frecuencia (o periodo), con contraste igual a 1, entonces, el contraste de la imagen (que también será cosenoidal), será el valor de la MTF para la frecuencia del objeto.

Si tenemos en cuenta que cualquier objeto puede ser descompuesto en una suma de cosenos de distinta frecuencia y dirección, entonces vemos que la MTF contiene la información de cómo varía el contraste de la imagen de cualquier objeto en función de la frecuencia espacial.

Se denomina frecuencia de corte del sistema óptico a aquella frecuencia para la cual el contraste en la imagen es 0, es decir, para la cual el valor de la MTF es nulo. Esta frecuencia de corte puede depender de la dirección.

Es necesario señalar que la MTF sólo nos aporta información del contraste. Para obtener una información completa de la imagen, nos falta otro elemento: la fase. Es decir, cómo varían las posiciones de los máximos y mínimos de cada uno de los cosenos en los que se descompone el objeto, sin por ello variar su frecuencia.

Esta información la contiene la denominada Función de Transferencia de Fase (PTF). A su vez, tanto la MTF como la PTF están contenidas en la Función de Transferencia Óptica (OTF), ya que,

$$OTF = MTFe^{iPTF}$$

Es decir, la MTF es el módulo de la OTF mientras que la PTF es su fase. Un sistema óptico puede caracterizarse tanto por la PSF como por la OTF, estando relacionadas entre sí por medio de una Transformada de Fourier.

A continuación definiremos la función que calcula la MTF sabiendo la PSF del sistema, así como la función de representación de la MTF, funciones que utilizaremos en las cuestiones planteadas.

In []:
```
# SOLO EJECUTAR
######################################################################
# MTF
######################################################################

# Calcula la FT, la centra, y su modulo es la MTF
def calcula_MTF(C,pupila):
    OTF=fftshift(fft2(calcula_PSF(C,pupila)))
    MTF=abs(OTF)
    return MTF/MTF.max() # Normalizamos la MTF para que el maximo valga 1


# Pinta la MTF
def dibuja_MTF(mtf,lambdam,zi,pupila_radio,npixels):
    # Define las frecuencias espaciales
    fxadim=linspace(-0.5,0.5,npixels) # frecuencias adimensionales (en 1/pixels) desde -0.5 a 0.5
    fx=fxadim/(dx*pupila_radio) # frecuencias con dimensiones (en 1/m)
    fy=fx
    # Nuevas coordenadas espaciales
    xnew=fx*lambdam*zi
    ynew=xnew
    dxnew=xnew[2]-xnew[1] # tamaño del pixel
    [Xnew,Ynew]=meshgrid(xnew,ynew); # coordenadas definidas en la matriz
    ffx=fxadim/dxnew # frecuencias con dimensiones (en 1/m)
    ffy=ffx
    figmtf=figure(figsize=(14,6))
    subplot(1,2,1)
    imshow(mtf,extent=(ffx[0]*1e-6,ffx[-1]*1e-6,ffy[0]*1e-6,ffy[-1]*1e-6),cmap='afmhot')
    xlabel('cpm')
    ylabel('cpm')
    title('MTF')
    colorbar()

    # Pinta perfil de la MTF
    pixelcentro=(npixels+1)/2; # pixel central
    mtfy=mtf[:,pixelcentro] # perfil MTF en y
    mtfx=mtf[pixelcentro,:] # perfil MTF en x

    subplot(1,2,2)
    plot(ffy*1e-6,mtfy,'k',ffx*1e-6,mtfx,'r')
    xlabel('ciclos/micra')
    ylabel('MTF perfil')
    legend(('en y','en x'))
    return 'ok'
```

Así, si queremos visualizar la MTF partiendo de unos coeficientes de la expansión en polinomios de Zernike

dados, seguiríamos los siguientes pasos

```
In []: # SOLO EJECUTAR
       coefs = loadtxt('ejemplo_Zernike_psf.dat')
       psf = calcula_PSF(coefs,pupila) # solo necesaria si se desea visualizar la PSF del sistema
       dibuja_PSF(psf,lambdam,zi,pupila_radio,npixels) # solo necesaria si se desea visualizar la PSF del sistema
       mtf = calcula_MTF(coefs,pupila)
       dibuja_MTF(mtf,lambdam,zi,pupila_radio,npixels)
```

Ejercicio 3

1. Dibujar la MTF para una aberración de onda caracterizada por un valor de S = 0.1 D (esfera), suponiendo un valor nulo para los coeficientes del desarrollo de Zernike no relacionados con S.
2. Repetir el apartado anterior para S = 0.3 D. ¿Qué ha provocado en la MTF un aumento de S?. ¿Cómo cambia la frecuencia de corte del sistema?
3. Dibujar la MTF para una aberración de onda caracterizada por un valor de C = 0.1 D (cilindro), y por un valor de $\theta = 24^o$ (eje), suponiendo un valor nulo para los coeficientes del desarrollo de Zernike no relacionados con C ni con θ. ¿Cómo cambia la frecuencia de corte del sistema comparada con la obtenida en el apartado 1?

MTF y contraste

La MTF nos da la información de cómo varía el contraste de la imagen dada por el sistema cuando varíamos la frecuencia espacial del objeto. Para ver esta propiedad, vamos a cargar en el documento un objeto con únicamente una frecuencia espacial, es decir, un objeto en donde la intensidad varía cosenoidalmente. A continuación calcularemos la imagen que el sistema proporciona de dicho objeto y calcularemos el contraste por inspección directa. Finalmente, compararemos el valor dado por la MTF con el contraste medido directamente de la imagen.

El siguiente código muestra el objeto y la imagen que proporciona el sistema óptico junto a dos perfiles espaciales que nos ayudan a calcular el contraste.

```
In []: # SOLO EJECUTAR
       objeto = imread('cos_frec_005lr.jpg',flatten=True)
       # A continuación, la variable que modifica la distancia a la
       # que se encuentra el objeto
       ##########
       dist = 3.0
       #########
       imagen_cos=calc_imagen(coefs,objeto,dist)

       figcut = figure(figsize=(14,3))
       ncut = shape(objeto)[0]/2
       perfilobjeto = objeto[ncut,:]

       num_pixeles_x = objeto.shape[0]
       num_pixeles_y = objeto.shape[1]
       pixel_tam_x = objeto_tam/num_pixeles_x
       pixel_tam_y = objeto_tam/num_pixeles_y
       x_obj = arange(num_pixeles_x)*pixel_tam_x*1e3 # en mm
       y_obj = arange(num_pixeles_y)*pixel_tam_y*1e3 # en mm

       thetarad=objeto_tam/dist #angulo en radianes
       # angulo refractado
```

```
thetatrans=thetarad/indice
# tamagno en vertical (en metros) de la imagen paraxial
imagen_tam=zi*tan(thetatrans)
pixel_tam_imag = imagen_tam/num_pixeles_y
y_im = arange(num_pixeles_y)*pixel_tam_imag*1e3 # en mm

ncut = shape(imagen_cos)[0]/2
perfilimagen = imagen_cos[ncut,:]
subplot(121)
plot(y_obj,perfilobjeto)
xlim(0,y_obj[-1])
xlabel('mm')
ylabel('perfil objeto')

subplot(122)
plot(y_im,perfilimagen)
xlim(0,y_im[-1])
xlabel('mm')
ylabel('perfil imagen')


plugins.connect(figcut, plugins.MousePosition(fontsize=14))
```

Ejercicio 4

1. Para una distancia de 2 m, calcular la frecuencia espacial del objeto en ciclos/grado, así como su contraste (utilizar la inspección directa en las gráficas superiores para ver el contraste del objeto).
2. Ejecutar la celda que se proporciona para obtener mediante inspección de la gráfica del perfil de la MTF su valor a la frecuencia del objeto.¿Con qué se relaciona este valor?.
3. Repetir los dos puntos anteriores para una distancia igual a 3 m.

Nota: La distancia se fija en el código mostrado en la celda superior a través de la variable `dist`.

In []:
```
#Indica aquí las operaciones para hallar la frecuencia espacial del objeto cosenoidal en ciclos/grado
```

In []:
```
# SOLO EJECUTAR
# Celda para dibujar el perfil de la MTF (Apartado 2 del Ejercicio 4)
mtf = calcula_MTF(coefs,pupila)
dibuja_MTF(mtf,lambdam,zi,pupila_radio,npixels)

# Pinta perfil de la MTF
pixelcentro = (npixels+1)/2; # pixel central
mtfy=mtf[:,pixelcentro] # perfil MTF en y
mtfx=mtf[pixelcentro,:] # perfil MTF en x

fxadim=linspace(-0.5,0.5,npixels) # frecuencias adimensionales (en 1/pixels) desde -0.5 a 0.5
fx=fxadim/(dx*pupila_radio) # frecuencias con dimensiones (en 1/m)
fy=fx
# Nuevas coordenadas espaciales
xnew=fx*lambdam*zi
ynew=xnew
dxnew=xnew[2]-xnew[1] # tamaño del pixel
[Xnew,Ynew]=meshgrid(xnew,ynew); # coordenadas definidas en la matriz
ffx=fxadim/dxnew # frecuencias con dimensiones (en 1/m)
ffy=ffx
```

```
fxcpg = ffx*zi*pi/(180*indice)
fycpg = fxcpg

figmtfprof = figure(figsize=(14,3))
plot(fycpg,mtfy,fxcpg,mtfx)
#xlim(0,0.001)
xlabel('ciclos/grado')
ylabel('MTF perfil')
legend(('en y','en x'));
plugins.connect(figmtfprof, plugins.MousePosition(fontsize=14))
```

Propagación de radiación en tejidos

Miguel Ángel Antón Revilla

1 Propagación de la radiación en tejidos biológicos

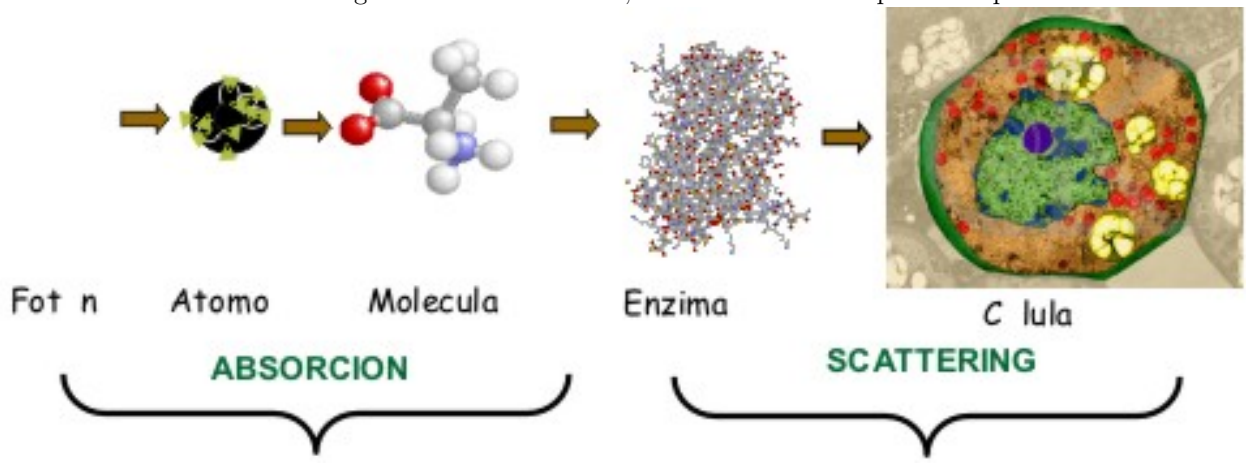
Los tejidos biológicos son medios inhomogeneos formados por partículas de diferentes tamaños y por una grna variedad de moléculas que absorben la radiación incidente sobre el tejido. En cualquier experimento óptico para obtener información biomédica, normalmente se ilumina el tejido con radiación conocida y se analizan los cambios experimentados esta radiación, bien en su contenido espectral, en su polarización, etc. Para obtener información del tejido será necesario: 1. Conocer las moléculas absorbentes que están en el tejido,que nos permita formular un modelo de la abasorción. 2. Tener una idea de los tamaños de los constituyentes que permitan formular un modelo de scattering. 3. Disponer de un modelo de la propagación de la radiación en el seno del tejido.

1.1 1.1 Procesos de absorción y scattering en medios biológicos

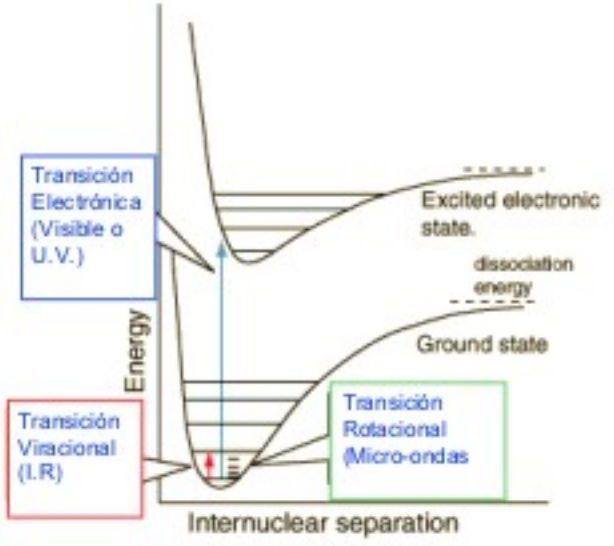
En óptica biomédica, la absorción es uno de los procesos más importantes:

1. La absorción permite tratamientos terapéuticos al transferir la energía de la radiación al tejido.
2. La absorción por moléculas permite desarrollar técnicas de diagnóstico basadas en análisis espectroscópico de los tejidos.
3. La absorción selectiva por determinadas moléculas del tejido suministra el contraste necesario para obtener imágenes que de otra forma no sería posible observar. Por ejemplo, los tintados de tejido permiten la discriminación celular al microscopio o la fluorescencia natural o inducida permite la creación de imágenes en vivo por debajo del tejido.

A nivel microscópico, los procesos de absorción ocurren cuando la frecuencia del fotón incidente sobre un medio material coincide con la separación energética entre los niveles de energía de los átomo, o moléculas que componen ese medio.



Mientras que los átomos presentan niveles de energía discretos, las moléculas presentan estructuras de niveles más complicadas debido a que no sólo hay movimientos electrónicos, sino que además los núcleos pueden vibrar en torno a la posición de equilibrios o rotar. Ello da lugar a la aparición de niveles electrónicos, vibracionales y rotacionales: En la figura se muestyra un diagrama típico de los niveles de una molécula. La figura de la izquierda es otro esquema que indica los niveles vibracionales de cada uno de los niveles electrónicos de la molécula (nivel fundamental S_0, primer nivel electrónico excitado o singlete S_1 con sus niveles vibracionalescorrespondientes, nivel escitado tipo triplete, T_1, etc. Este diagrama se denomina, diagrama de Jablonsky. de los átomos son niveles discretos.



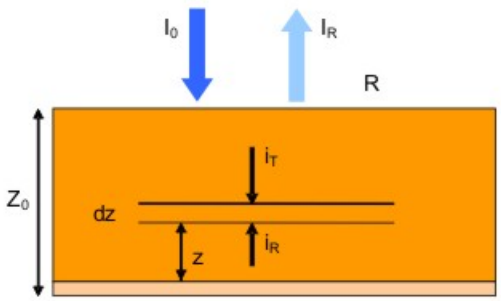
Las moléculas pueden experimentar más tipos de movimientos, tales como vibracion de los nucleos o rataciones. Por ello la estructura de niveles se complica y los espectros de absorción y emisión tambien, dado lugar a la aparición de diferentes tipod de noveles de energía, electrónicos, vibracionales y rotacionales, entre otros.

2 Propagación de la radiación en tejidos biológicos

Los tejidos biológicos son, en general, medios altamente inhomogéneos. Por ello, la radiación experimenta múltiples eventos de absorción y scattering. En cada proceso de scattering, los fotones cambian de dirección dando lugar a trayectorias aleatorias. Desde un punto de vista macroscópico, la radiación se difunde dentro del medio. Existen diferentes maneras de describir la difusión de la radiación. Unos se basan en métodos probabilísticos como los modelos de Monte-Carlo. Otros son métodos analíticos basados en la ecuacion de difusión. Aquí haremos una introducción cualitativa a la difusión y obtendremos una serie de soluciones analíticas para el caso de iluminación estacionaria y pulsada.

2.1 2.1 Modelo de Kubelka-Munk

Consideremos un haz de radiación de intensidad I_0 que incide sobre una lámina de espesor z_0 y que presenta en la primera interfase, una reflectancia R. La interfase final presentaráa en general, una reflectancia R'. La radiación que se retro-difunde hacia arriba después de haber experimentado procesos de absorción y scattering es IR. Consideremos una capa infinitesimal de espesor dz situada a una distancia z de la parte inferior de la superficie. Sobre esta capa incide la radiaci´on esparcida por el medio hacia arriba y hacia abajo. Sea i_T la irradiancia que incide hacia abajo e i_R la que incide en la dirección hacia arriba de la capa.



Definiremos un coeficiente de absorción por unidad de longitud promedio K, y otro coeficiente que tenga en cuenta el scattering, S. El efecto de estos dos procesos sobre la capa infinitesimal se deben a los siguientes procesos:

1. Decrecimiento de iT debido a la absorción y scattering :−(S + K)iT dz
2. Decrecimiento de iS debido a la absorción y scattering :−(S + K)iRdz
3. Aumento de iT debido al scattering de iR: S iRdz
4. Aumento de iT debido al scattering de iS: S iT dz

Con todo ello, la variación de la intensidad antes y después de la capa se podrá escribir como

$$-di_T = -(S+K)i_T dz + i_R S dz$$
$$di_R = -(S+K)i_R dz + i_T S dz$$

Dividiendo la primera ecuación por por i_T y la segunda por i_R y restando ambas ecuaciones se llega a

$$\frac{di_R}{i_R} - \frac{di_T}{i_T} = -2(S+K)dz + S\left(\frac{i_T}{i_R} + \frac{i_R}{i_T}\right)dz$$

Si definimos el coeficiente $r = \frac{i_R}{i_T}$, la ecución anterior se puede poner como

$$d(ln\frac{i_R}{i_T}) = \frac{dr}{r} = \left[-2(S+K) + S\left(r + \frac{1}{r}\right)\right]dz$$

Reagrupando los términos que dependen de r e integrando se llega a

$$\int_{R'}^{R} \frac{dr}{r^2 - 2\left(1 + \frac{K}{S}\right)r + 1} = \int_0^1 S dz$$

donde R y R' representan las reflectancias en las dos caras de la lámina.

Si introducimos las definiciones $a = 1\frac{K}{S}$ y $b = \sqrt{a^2 - 1}$, esta integral conduce a

$$ln\left(\frac{R-a-b}{R-a+b}\right)\left(\frac{R'-a+b}{R'-a-b}\right) = 2bS$$

Vamos a considerar ahora el caso particular de que el espesor de la lámina sea infinito. En ese caso, la reflectancia del la lámina la denominaremos R_∞. Por otro lado la si el espesor de la lámina tiende a infinito, la reflectancia R' se debe anular. Entonces se debe anular tambien el denominador de la ecuación anterior, es decir:

$$R = 1 + \frac{K}{S} - \sqrt{(\frac{K}{S})^2 + 2\frac{K}{S}}$$

2.2 2.2 Modelo difusivo. Ecuación de la difusión

En esta sección vamos a presentar un modelo difusivo cualitativo, que mejora al modelo de Kubelka-Munk.

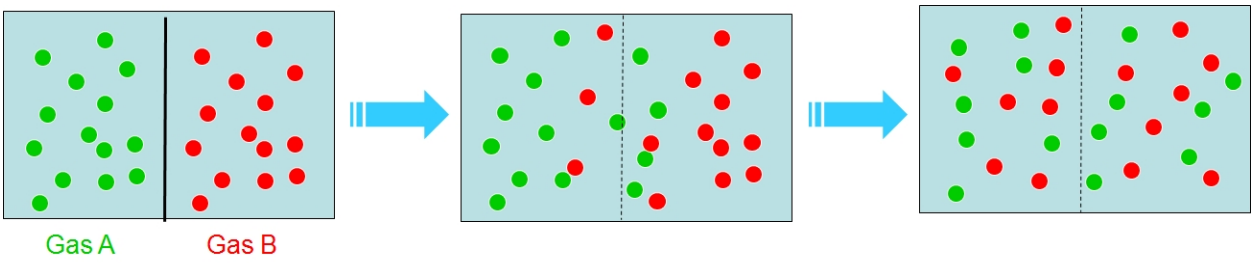


Figure 1: difusion1

En la figura se muestra el proceso de difusión de dos gases A y B que inicialmente estaban separados. Se puede observar que hay un flujo de moléculas en ambas direcciones hasta que se llega a una distribución homogénea. Si ahora consideramos que los fotones encerrados en una región de la caja se comportan como un gas de fotones, y eliminamos la separación, se producirá una corriente desde la zona de mayor concentración a la de menor concentración. Para describir matemáticamente este flujo introduciremos dos magnitudes, la densidad de energía **U(z,t)** y la intensidad de corriente **I(z,t)** definidas como

$$U(z,t) = \frac{N_{particulas}}{Volumen} h\nu$$

$$I(z,t) = \frac{N_{particulas}}{Areaxtiempo} h\nu$$

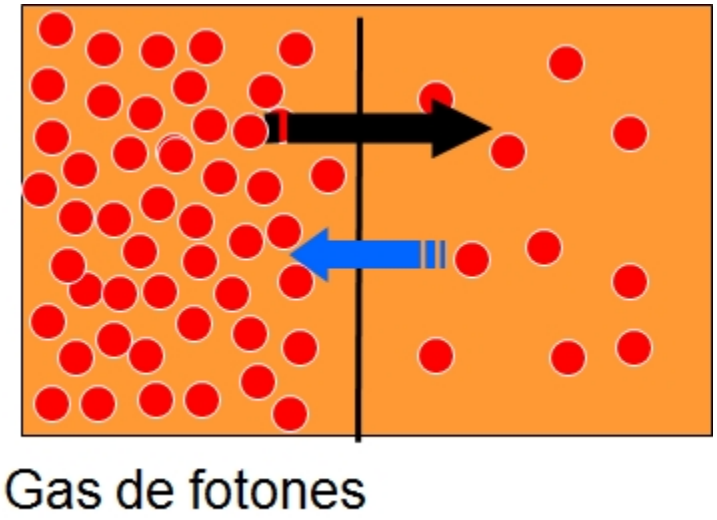


Figure 2: difusion1

La primera ley de la difusión, denominada **ley de Fick**, establece que la variación de la intensidad que fluye a través de una superficie es

$$I_d(z,t) = -d\frac{\partial U(z,t)}{\partial z}$$

,

donde **d** es una constante de proporcionalidad. Supongamos ahora un volumen elemental dentro del medio como se indica en la figura.

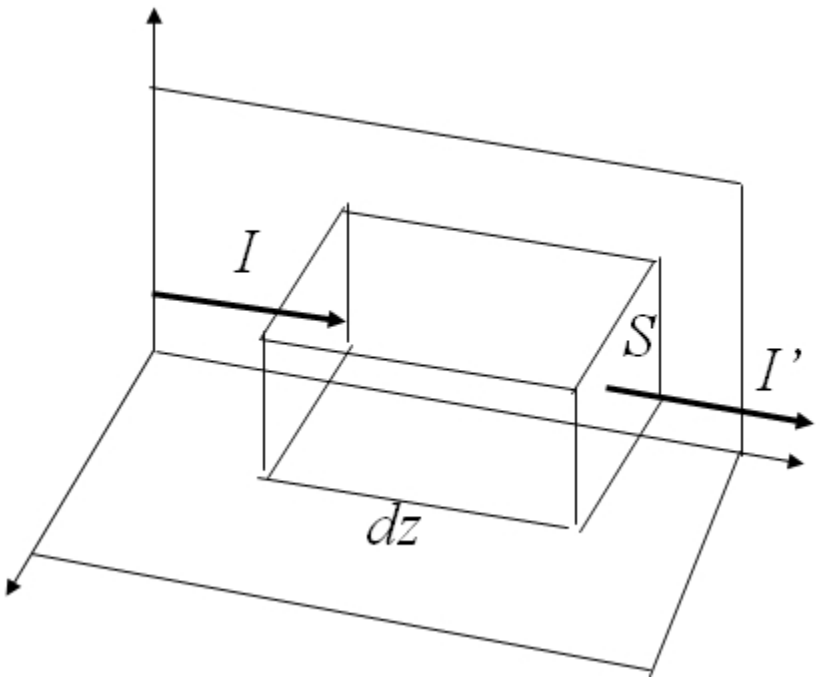


Figure 3: difusion1

En el volumen elemental de área **A** y longitud **dz**, el incremento de partículas por segundo en el volumen V, vendrá dado por la diferencia entre el flujo entrante y el flujo saliente:

$$\left[I(z,t) - I(z+dz,t)\right] S \simeq -\frac{\partial I(z,t)}{\partial z} dz A$$

Por otra parte, este incremento se podrá obtener en términos de la densidad de partículas por unidad de volumen **U(z,t)** como:

$$\frac{\Delta N_{particulas}}{\Delta t} = \frac{\partial U(z,t)}{\partial t} A dz$$

Igualando ambos términos, obtenemos

$$\frac{\partial U(z,t)}{\partial t} = -\frac{\partial I(z,t)}{\partial z}$$

Si ahora, tenemos en cuenta la ley de Fick, podemos obtener una ecuación sólo en términos de la densidad de energía:

$$\frac{\partial U(z,t)}{\partial t} = d\frac{\partial^2 U(z,t)}{\partial z^2}$$

Hasta ahora solo hemos considerado la propagación sin tener en cuanta la posibilidad de que en el medio haya partículas absorbentes con coeficiente μ_a y que además reciba energía desde el exterior al ser iluminado con una fuente que emite una densidad de energía **S(r.t)**. En ese caso deberemos incorporar la energía por unidad de volumen absorbida por el medio $\mu_a I(r,t)$ que aparecerá como una fuente de pérdidas de energía del haz de radiación y la energía aportada por la fuente:

$$\frac{\partial U(z,t)}{\partial t} = d\frac{\partial^2 U(z,t)}{\partial z^2} - \mu_a I(r,t) + S(r,t)$$

Finalmente, si tenemos en cuenta la relación entre **U(r,t)= I(r,t)/c** donde **c** es la velpcidad de la luz en el medio, se puede obtener una ecuación solo en términos de **U(r,t)** o de **I(r,t)**:

$$\frac{1}{c}\frac{\partial I(z,t)}{\partial t} = D\frac{\partial^2 I(z,t)}{\partial z^2} - \mu_a I(r,t) + S(r,t)$$

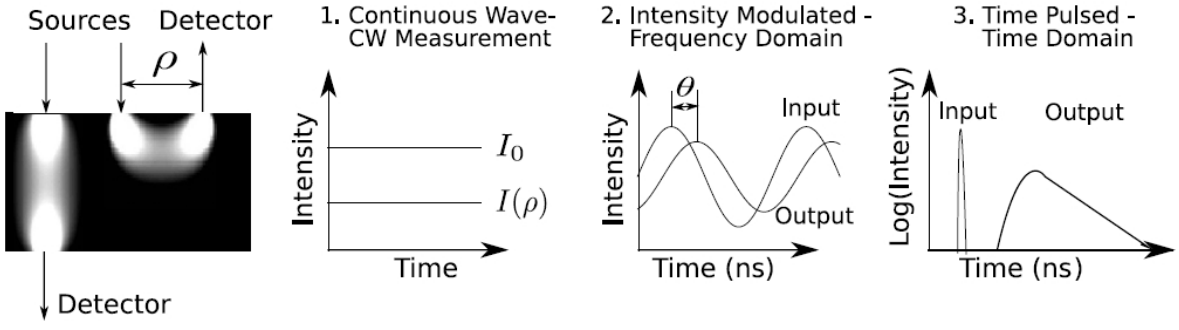
donde $D = \frac{d}{3}$ es el coeficente de difusión. Una teoría de la obtención de la ecuación de difusión más rigurosa permite obtener el valor de D como:

$$D = \frac{1}{\sqrt{3\mu_a(\mu_a + \mu_s')}}$$

Este coeficiente tiene dimensiones de longitud, y representa el recorrido medio del fotón entre do eventos de scattering en un medio difusivo.

2.3 2.3 Soluciones de la ecuación de difusión

En general la ecuación anterior se puede resolver por métiodos numéricos, pero en algunos casos es posible obtener soluciones analíticas de interés. Existen tres situaciones típicas que que se corresponden con el tipo de fuente que ilumina en medio biológico. En la figura se presentan los diferentes casos: (a) iluminaciñon estacionaria, en la cual, la intensidad de la fuente no varía en le tiempo, (b) iluminación modulada, en la que se hace variar la ilumnacíon de forma periódica controlada, y (c) iluminación pulsada, en la que el medio recibe un pulso de luz, típicamente con pulso de nano o femtosegundos. Dependiendo del tipo de fuente, obtendremos diferentes soluciones, que vamos a nalizar a continuación.



2.4 2.3.1 Iluminación estacionaria en un medio infinito.

Vamos a considerar que se ilumina el tejido con una fuente puntual, cuya densidad de energía no depende del tiempo, es decir, $S(r,t) = S0$. En ese caso la ecuación de difusión obtenida más arriba no depende del tiempo y se reduce a una ecuación más simple:

$$D\frac{\partial^2 I(z)}{\partial z^2} - \mu_a I(z) - S_0 = 0$$

Es más usual re-escribir esta ecuación como

$$I(z) - \frac{1}{\mu_{eff}^2}\frac{\partial^2 I(z,t)}{\partial z^2} = \frac{S(r,t)}{\mu_a} = 0$$

donde $\mu_{eff} = \frac{\mu_a}{D}$ es el coeficente efectivo. La solución de esta ecuación en un medio infinito es:

$$I(r) = \frac{I_0}{4\pi D r} e^{-\mu_{eff} r}$$

Vamos a representar esta solucion para unos valores dados del coeficiente de absorcion y de scattering que se pueden cambiar:

Ejercicio 1

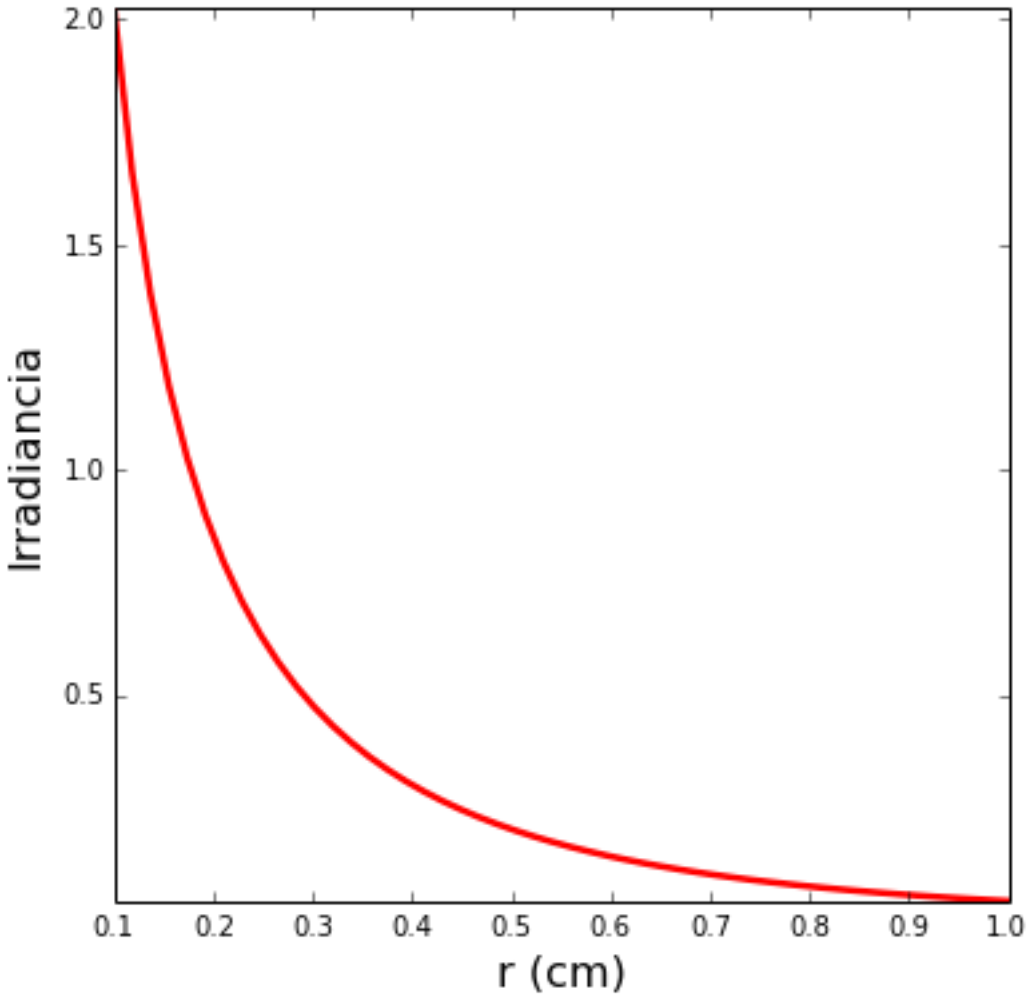
Representar la variación de irradiancia con la distancia producida por una fuente de 0.1 W de potencia en un medio que presenta un coeficente de absorción $\mu_a = 0.1cm^{-1}$ y un coefiente de scattering reducido $\mu_s' = 10cm^{-1}$

In [43]:
```
%pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
#Parámetros a modificar
#- - - - - - - - - -
mua=0.1;        # coeficiente de absorción en  'cm^{-1}'
musprima=10.0;  # coeficiente de scattering en 'cm^{-1}'
#- - - - - - - - - -
rf=1.0;    #---> especificado en 'cm'
I0=0.1;    #---> especificado en 'W'
D = 1.0/(3.0*(mua+musprima));       #coeficiente de difusión
mueff=sqrt(3.0*mua*(mua+ musprima));#coeficiente efectivo
r =linspace(0.1,1,50);              #variable espacial en 'cm'
I = I0/(4.0*pi*D*r)*(exp(-mueff*r));#intensidad de corriente
#- - - - - - - - - -
#Calculamos los valores mínimo y máximo de los ejes de la figura
minX=min(r);
maxX=max(r);
minY=min(I);
maxY=max(I);
#- - - - - - - - - -
#Realizamos el gráfico
figure(figsize=(6,6));
plot(r,I,linewidth=2.5,color='r');
xlabel('r (cm)',fontsize=16);  #Ponemos las etiquetas en los ejes
ylabel('Irradiancia',fontsize=16);
axis([minX, maxX, minY, maxY]);#Establecemos los límites en los ejes
```

Out[2]: [0.10000000000000001, 1.0, 0.0422922617929814, 2.0259819846130656]



Ejercicio 2

Se envía un pulso laser de 1W de potencia y duración 0.1 s a través de una fibra insertada en un gel que contiene tinta como medio absorbente y leche como medio de scattering. Las propiedaes ópticas del gel son $\mu_a = 0.1cm^{-1}$, y $\mu_s' = 10cm^{-1}$. La longitud de penetración de la luz que envía la fibra viene dada por $z_0 = 1/(\mu_a + \mu_s')$ justo en frente de la fibra. Podemos considerar que toda la luz se concentra en un punto a esta distancia de la fibra y lo tomaremos como origen de coordenadas, $r = 0$. Se sitúa una esfera absorbente de 100 micras de diámetro centrada a 5 mm en frente de la fuente. El coeficiente de absorción de la esfera es $\mu_a = 10cm^{-1}$. Estimar el aumento de temperatura que tiene lugar en la esfera al final del pulso. Ignorar la distribución de temperatura en el interior de la esfera y la difusión térmica.

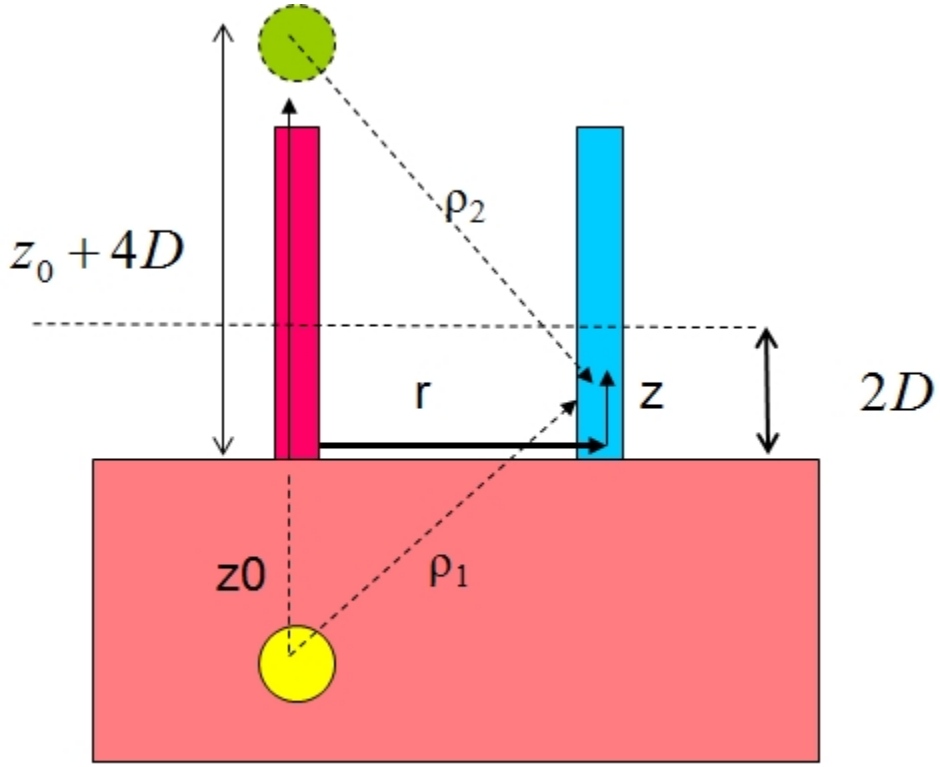
```
In [3]: mua = 0.1; # #Coeficiente de absorción de la esfera 'cm^-1'
        musp= 10;  # cm^-1
        r = 0.5;   # cm
        mua_esfera= 10;   #Coeficiente de absorción de la esfera 'cm^-1'
        D       = 1/(3*(mua+musp));
```

```
mueff=sqrt(3*mua*(mua+ musprima));#Calculamos el coeficiente efectivo
P_0 = 1;  # Potencia incidente en 'W'
P = P_0/(4*pi*D*r)*(exp(-mueff*r)) ;
t = 0.1;  # s
Q = P*t;  # Energía transferida en la posición de la esfera
dTemp   = Q/4.18;
print "P=",P
print "Q=",Q
print "dTemp=",dTemp
```

```
P= 2.01965334809
Q= 0.201965334809
dTemp= 0.0483170657438
```

2.5 2.3.2 Iluminación estacionaria en un medio semi-infinito.

En general, en un experimento real, el medio material presenta al menos una frontera de separación definida entre el tejido y el medio exterior. En este caso se trata de un medio semi-infinito. Para calcular la reflectancia y la transmitancia difusas deberemos incorporar ciertas condiciones de contorno y construir una solución a partir de la solución anterior para un medio infinito, que cumpla estas condiciones de contorno o frontera. En efecto, consideremos la situación de la figura:



Para empezar, la radiación enviada por la fibra excitadora se puede considerar que se concentra en un punto situado a una distancia igual a la longitud de penetración efectiva, $z_0 = \frac{1}{\mu_a + \mu_s'}$. Además, para el caso de un medio semi-infinto la condición de contorno se reduce a (ver apuntes en campus virtual)

$$I(z-4d,t)=0$$

esto es, la intensidad se debe anular en todos los puntos de una frontera plana situada a una distancia **2D** de la frontera real. Parece claro que con una sola fuente en el interior del medio, es imposible satisfacer la condición de contorno en la frontera. Para ello situamos una fuente virtual de intensidad negativa $-I_0$ situada a la misma distancia de la frontera que la fuente real, esto es a z_0+4D respecto del origen $z=0$. Con ello, la intensidad en un punto cualquiera de la frontera sera nula. La intensidad en cualquier otro punto cualquiera se obtendrá sumando las contribuciones de las dos fuentes en el punto considerado:

$$I(r)=I_1(z_0,r,\rho_1)+I_1(z_0,r,\rho_2)=\frac{I_0}{4\pi Dr}\left[\frac{e^{-\mu_{eff}\rho_1}}{\rho_1}-\frac{e^{-\mu_{eff}\rho_2}}{\rho_2}\right]$$

donde ρ_1 y ρ_2 son las distancia desde cada fuente al punto del medio considerado. estas distancias valen:

$$\rho_1=\sqrt{z_0^2+r^2}$$
$$\rho_2=\sqrt{[(z_0+4D-z)^2+r^2]}$$

Una vez obtenida la solución general que satisface las condiciones de contorno, podemos calcular la reflectancia sin más que aplicar la primera d ley de Fick:

$$R_d(z,t)=-\frac{1}{I_0}D\frac{\partial I(z,t)}{\partial z}=-\frac{1}{I_0}D\frac{\partial I(z,t)}{\partial \rho}\frac{\partial \rho}{\partial z}=$$

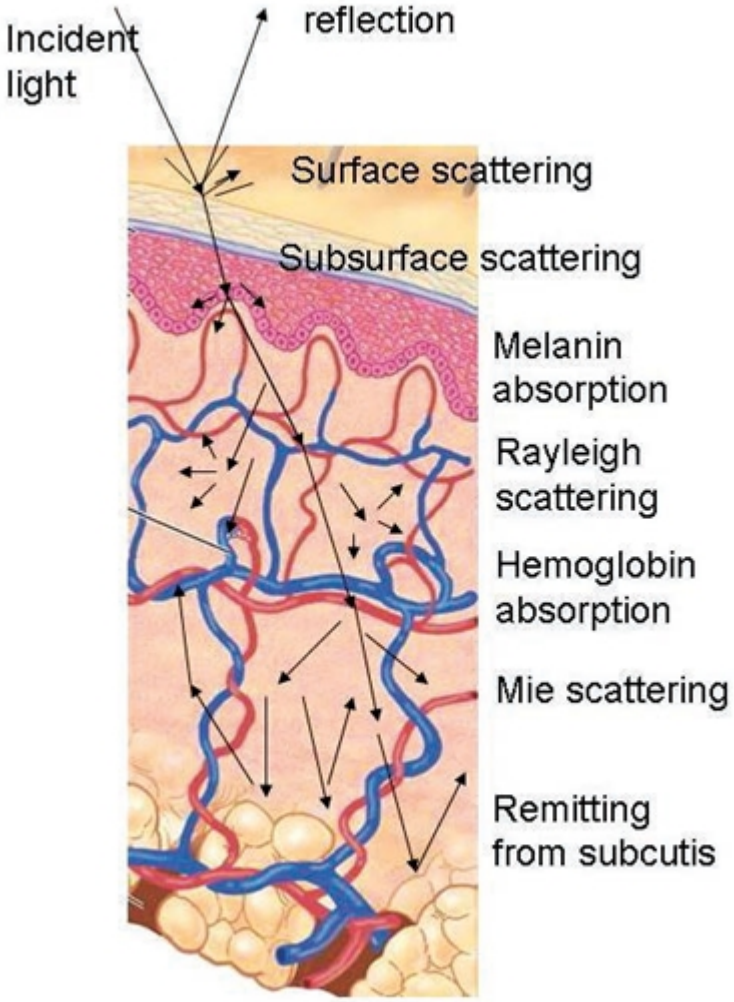
. El resultado de realizar estas derivadas es:

$$R_d(\rho_1,\rho_1,r,z_0)=\frac{1}{4\pi D}\left[z_0\frac{(1+\mu_{eff}\rho_1)e^{-\mu_{eff}\rho_1}}{\rho_1^3}+(z_0+4D)\frac{(1+\mu_{eff}\rho_2)e^{-\mu_{eff}\rho_2}}{\rho_2^3}\right]$$

2.6 2.3.3 Aplicación a la caracterización de un tejido biológico: la piel.

Como aplicación del modelo de difusión que hemos desarrollado más arriba, estudiaremos la obtención de información de concentración de cromóforos en el caso de la piel. Este medio es importante dado que para muchas medidas no invasivas realizadas sobre el cuerpo humano, deberemos iluminar la zona de interés a través de la piel. La piel humana tienen dos principales fuentes de scattering: scattering de superficie y scattering de sub-superficie. La superficie de scattering está influenciada por irregularidades en la capa corneal (deshidratación de la piel, pecas...) y es similar a la reflexión. El scattering en la sub-superficie tienen lugar en las membranas y corpúsculos celulares tales como los queratocitos, melanosomas, organelas celulares, mitocondria, núcleos celulares, agregados de proteínas y fibras de colágeno. La capa corneal y la epidermis se caracterizan como medios de scattering unidireccionales debido a la orientación de las fibras, mientras que las estructuras celulares dan lugar a scattering Mie. Las largas fibras de colágeno en la retícula de la dermis producen scattering Mie dirigido hacia adelante, mientras que el scattering Rayleigh tiene lugar en la pequeñas estructuras de fibrillas de colágeno en la dermis papilar y otras microestructuras. Las fibras de colágeno constituyen una de las principales fuentes de scattering no solo en la piel sino también en la esclera del ojo. El colágeno produce scattering a nivel macroscópico a travéss de las fibras de colágeno cuyo tamaño varía entre 2 y 8 μm. Este scattering es de tipo Mie. Además de las fibras macroscópicas de colágeno, también aparecen ultraestructuras de fibrillas de colágeno que presentan estriaciones periódicas con periodos del orden de 70 nm. Estas fibrillas están constituidas de moléculas de tropocolágeno. La fluctuaciones periódicas del índice de refracciónn en estas ultraestructuras contribuyen al scattering Mie, y

dado su pequeño tamaño, comparado con la longitud de onda en el visible, se denomina componente Rayleigh



del scattering Mie.

Por lo tanto, debido a que el tejido biológico está constituido por partículas de diferentes tamaños y medios de diferentes índices, para modelizar el coeficiente de scattering reducido se adopta un modelo que combina el scattering Rayleigh y Mie:

$$\mu_s'(\lambda) = a' \left[f_{Ray} \left(\frac{\lambda}{500nm} \right)^{-4} + (1 - f_{Ray}) \left(\frac{\lambda}{500nm} \right)^{-b_{Mie}} \right]$$

donde f_{Ray} representa la fracción de scattering Rayleigh en el coeficiente de scattering reducido, y b_{Mie} es un número que se suel denominar *potencia de scattering*. Por otra parte la absorción se suele formular asumiendo que el coeficiente de absorción es una combinación lineal de coeficientes de absorción $\mu_{a,j}$ de los cromóforos que constituyen el tejido, esto es:

$$\mu_a(\lambda) = \sum_j f_j \mu_a^j$$

donde f_j representa la fracciones en volumen de cada cromóforo. En el caso de la piel, los cromóforos más importantes son la hemoglobina, Oxihemoglobina y agua. Por lo tantom propondremos como modelo de absorción :

$$\mu_a(\lambda) = f_1 \left[S\mu_a^{Oxihem} + (1-S)\mu_a^{Hem} \right] + f_2\mu_a^{Agua}$$

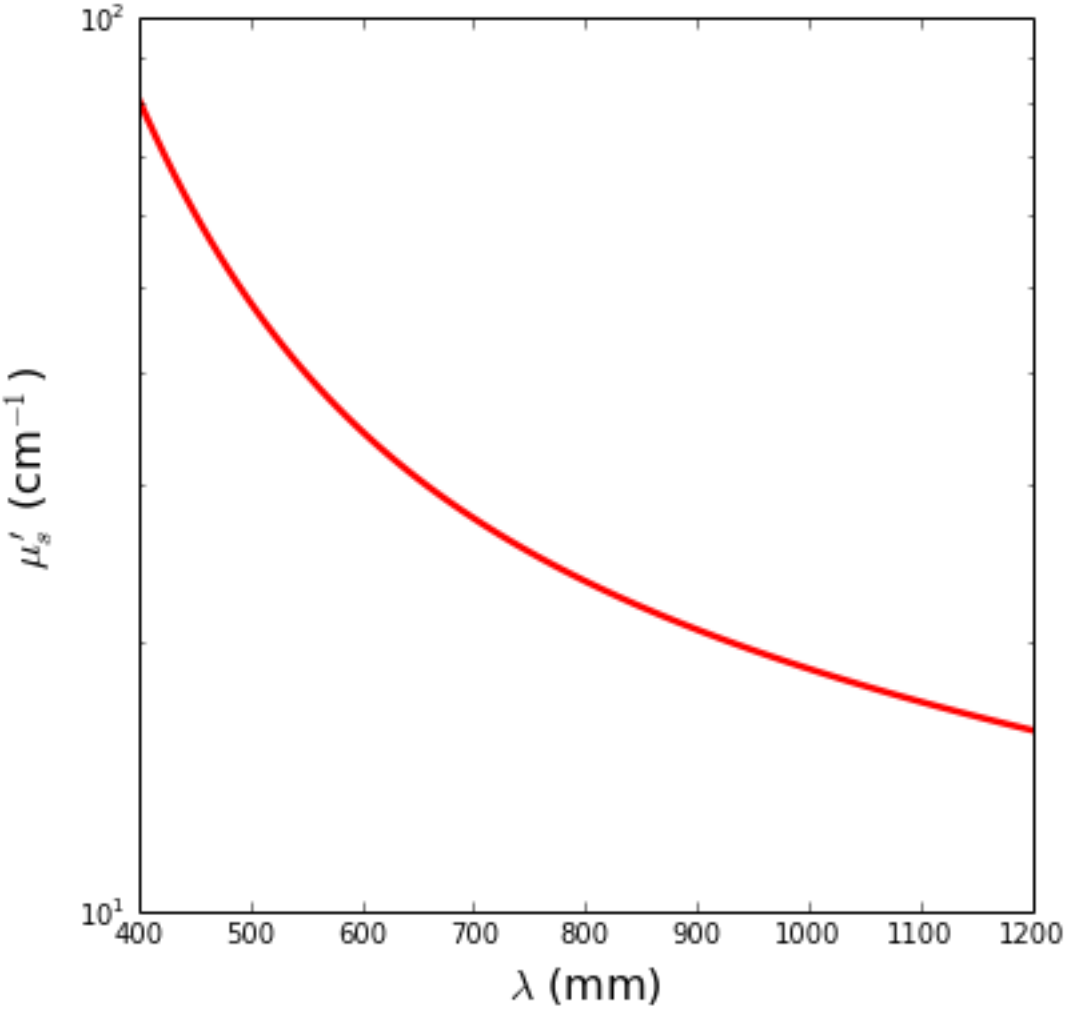
donde S representa la fracción de oxihemoglobina respectop de la fracción de sangre total, esto es el parámetro de oxígeno en sangre (venosa y arterial). Para más información, sobre las propiedades ópticas de tejidos biológicos se pueden consultar los siguientes artículos articulo1, articulo2 o la siguiente página web

Ejercicio 3

Representar el coeficiente de scattering de la piel y del tejido cerebral en función de la longitud de onda en el intervalo [400 1200] nm, teniendo en cuenta los datos de los coeficientes del modelo combinado que se dan en el artículo de revisión "articulo1". Nota: En el programa se deben cambiar los coeficientes **fRay** y **bMie** correspondientes a cada tejido para ver el efecto en el resultado final.

In [44]:
```
a=48;          # cm^{-1} ---> cambiar el coeficientes para el tejido correspondiente
fRay= 0.409;  #---> cambiar el coeficientes para el tejido correspondiente
bMie=0.702;   #---> cambiar el coeficientes para el tejido correspondiente
landa=linspace(400,1200,100);
muRay=a*(fRay*pow(landa/500,-4));
muMie=a*((1-fRay)*pow(landa/500,-bMie));
musprima=muRay+ muMie;
#- - - - - - - - - -
#Realizamos el gráfico
figure(figsize=(6,6));
semilogy(landa,musprima,'r',linewidth=2.5);
xlabel('$\lambda$ (mm)',fontsize=16);               # Etiqueta del eje X
ylabel('$\mu^{\prime}_s$ (cm$^{-1}$)',fontsize=16); # Etiqueta del eje Y
```

Out[44]:
```
<matplotlib.text.Text at 0x7f9df8559e90>
```



Ejercicio 4

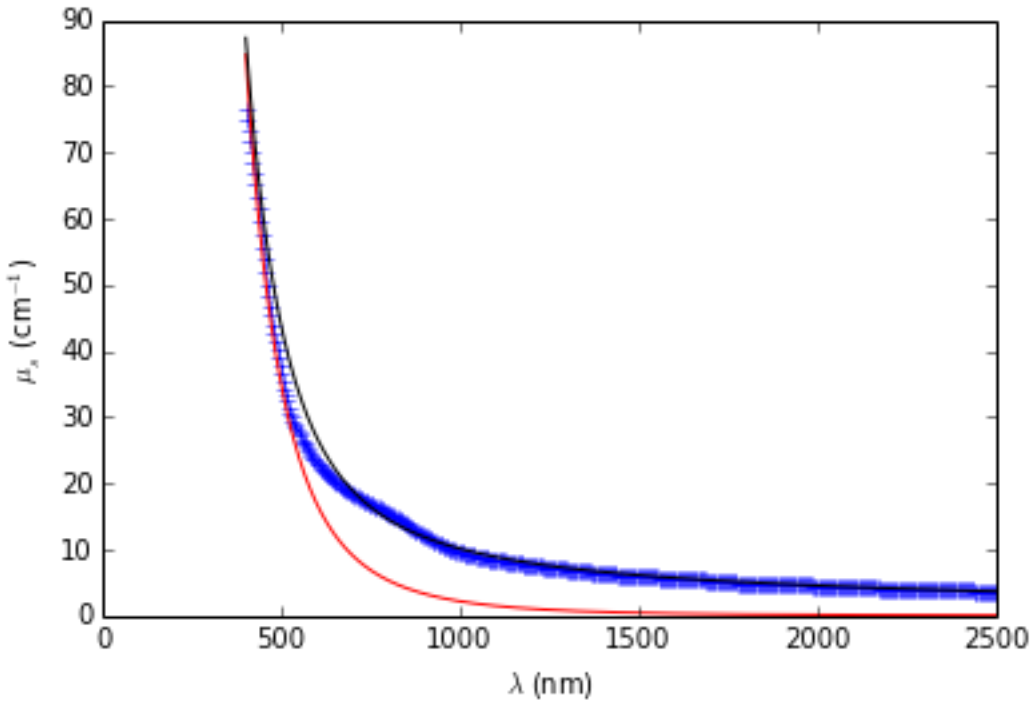
Se ha medido experimentalmente el scattering reducido de la piel y los resultados se dan en el fichero "scattering.dat". Se ha realizado un ajuste de los datos a un modelo que combina el scattering Rayleight y el scattering Mie. El resultado vienen dado por la expresión

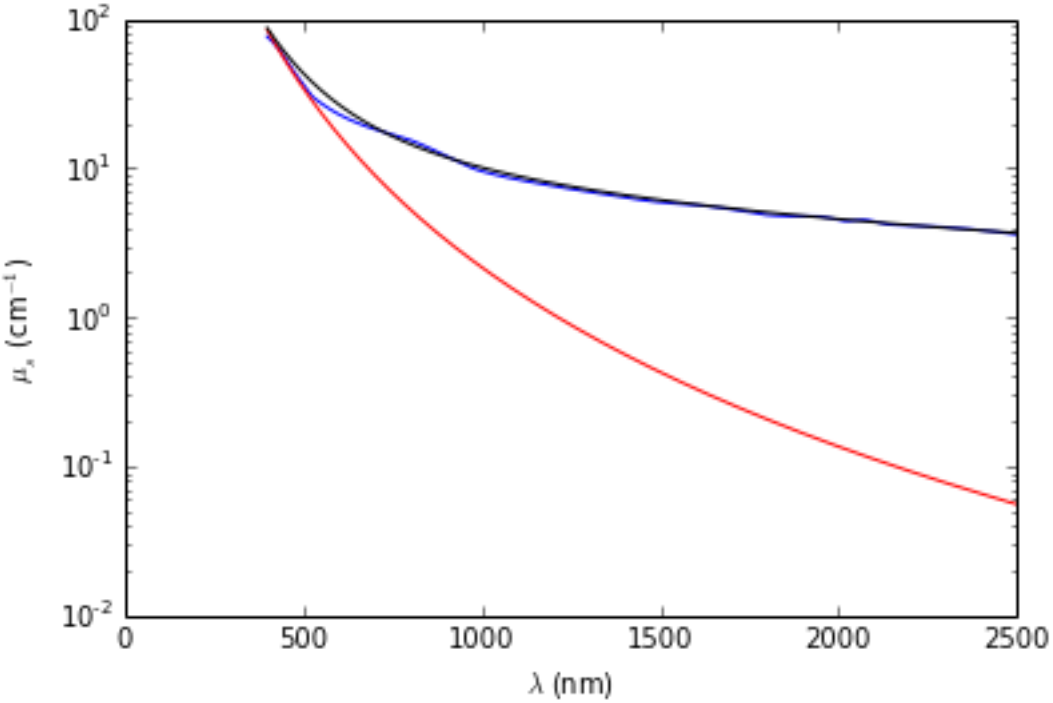
$$\mu_s'(\lambda) = \frac{2.17 \times 10^{12}}{\lambda^4} + \frac{4.59 \times 10^3}{\lambda^{0.93}}$$

donde la longitud de onda se expresa en nm. Representar en un mismo gráfico, los datos experimentales, la contribución del scattering Rayleigh, la del scattering Mie y el scattering total. Comentar las zonas donde es importante cada tipo de scattering.

```
In [45]: #MANERA DIRECTA DE CARGAR ESPECTROS
         scatteringpiel = loadtxt('scatteringpiel.dat')

         landa = scatteringpiel[:,0];
```

```
mu_piel=scatteringpiel[:,1];
mu_Rayleigh= 2.17e12/(landa)**4;

mu_Mie= 4.59e3/(landa)**(0.913);
mu_Total=4.59e3/(landa)**(0.913)+ 1.74e12/landa**4;
#-----------------------------------------------------------------------
figure()
plot(landa, mu_piel,'+',landa,mu_Rayleigh,'r',landa,mu_Total,'k' );
xlabel('$\lambda$ (nm)');
ylabel('$\mu_s$ (cm$^{-1}$)');

figure()
semilogy(landa, mu_piel,'b',landa,mu_Rayleigh,'r',landa,mu_Total,'k' );
xlabel('$\lambda$ (nm)');
ylabel('$\mu_s$ (cm$^{-1}$)');
```



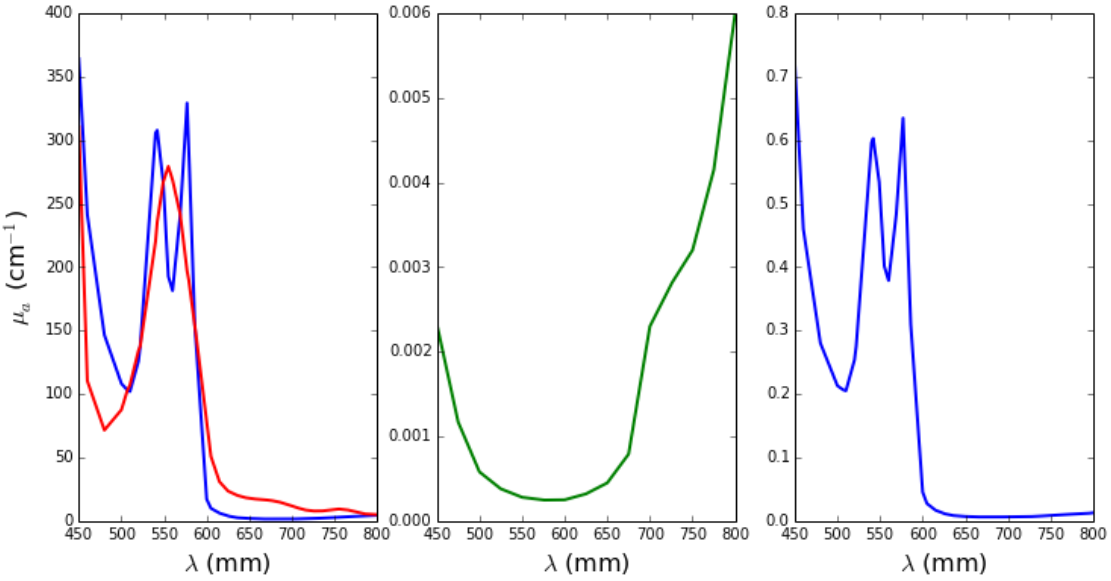


Ejercicio 5

Representar en un mismo gráfico el coeficiente de absorción lineal de la hemoglobina, la oxhiemoglobina y el agua. En otra gráfica representar el coeficiente de absorción lineal de la piel en función de la longitud de onda en el intervalo [400 800] nm, teniendo en cuenta que los cromóforos relevantes son la hemoglobina, la oxihemoglobina y el agua. Los ficheros de los cromóforos se dan en Hemoglobina.dat, Oxhihemoglobina.dat y AGUA.dat, respectivamente. Asumir que las fracción en volumen de sangre (hemoglobina+ Oxihemoglobina) es $f_1=0.35, y la fracción de agua es f_2=0.65. A su vez, la fracción de oxhihemoglobina es S=0.75.

```
In [46]: # load muHemoglobina.dat;
         muHemo=loadtxt('muHemoglobina.dat');
         landa= muHemo[:,0];
         mua_Hemo = muHemo[:,1];
         muOxih=loadtxt('muOxihemoglobina.dat');
         mua_Oxy=muOxih[:,1];

         AGUA=loadtxt('AGUA.dat');
         mua_Agua=AGUA[:,1];

         f1 =0.002;
         S = 0.91;
         RO= 0.504;
         f2=0.65;

         #calculamos el coeficiente total
         mua_total=f1*( S*mua_Oxy + (1-S)*mua_Hemo )+ f2*mua_Agua;
         figure(figsize=(12,6));
         subplot(1,3,1);
```

```
plot(landa, mua_Oxy,'b',landa, mua_Hemo,'r',lw=2);
#axis([400,800,0,350]);
#plot(landa, mua_Oxy,color='b',linewidth=2.5);
#plot(landa, mua_Hemo,color='r',linewidth=2.5);
#plot(landa, mua_Agua,color='g',linewidth=2.5);
xlabel('$\lambda$ (mm)',fontsize=16);
ylabel('$\mu_a$ (cm$^{-1}$)',fontsize=16);
#nuevo subplot
subplot(1,3,2);
plot(landa,mua_Agua,'g',lw=2);
xlabel('$\lambda$ (mm)',fontsize=16);
subplot(1,3,3);
plot(landa,mua_total,lw=2);
xlabel('$\lambda$ (mm)',fontsize=16);
```

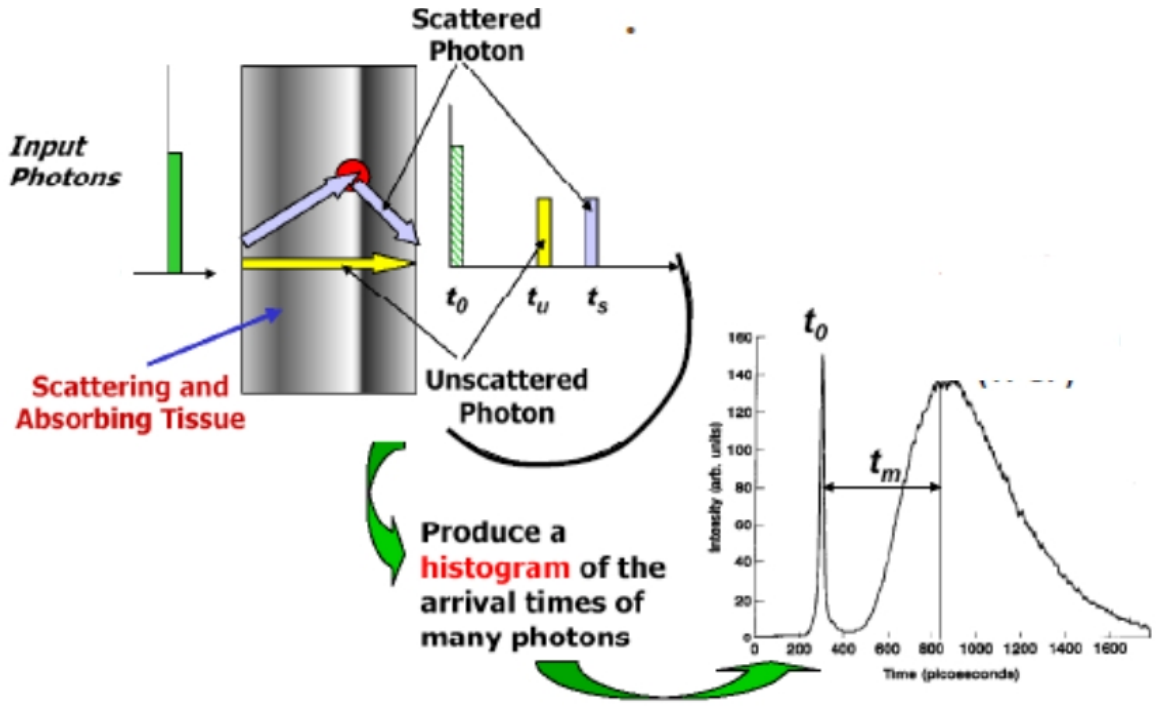


3 2.4 Reflectancia y transmitancia difusa transitoria

En el caso de medidas resueltas en el tiempo, se ilumina la mue stra con un pulso proce- dente de una fuente que emite pico o femto-segundos. Después de pasar por el tejido, bién en reflexión o transmisión, se miden los cambios temporales e xperimentados por el pulso con un fotomultiplicador o con una streak camera. Las características temporales del pulso esparcido o transmitido llevan información de las propiedades del tejido.

Vamos usar las soluciones obtenidas más arriba al resolver la ecuación de difusión, y los vamos a aplicar a una lámina plano-paralela de tejido homogéneo. Un pulso colimado de luz se dirige a una muestra de tejido y se usa un detector situado a cierta distancia para resolver temporalmente el pulso transmitido. La geometría el problema se muestra en la Fig. 3.73. El pulso de un haz láser colimado se hace incidir normalmente a una superficie semi-infinita [Fig. 3.73(a)] de tejido homogéneo. Asumiremos que la fluencia del pulso satisface la ecuación de difusión que se vió más arriba:

$$\frac{1}{c}\frac{\partial I(z,t)}{\partial t} = D\frac{\partial^2 I(z,t)}{\partial z^2} - \mu_a I(r,t) + S(r,t)$$



donde $D = \frac{d}{3}$ es el coeficente de difusión. Una teoría de la obtención de la ecuación de difusión más rigurosa permite obtener el valor de D como:

$$D = \frac{1}{\sqrt{3\mu_a(\mu_a + \mu_s')}}$$

La solución de esta ecuación para un pulso temporal cuasi-instantáneo es:

$$I(r,t) = \frac{ce^{-\mu_a ct}}{(4\pi Dct)^{3/2}} e^{-\frac{r^2}{4Dct}}$$

Para el caso de un tejido que presenta una frontera, deberemos incorporar la condición de contorno, tal como se hizo en el caso estacionario. En este caso, se tendrá:

$$I_d(\rho,z,t) = \frac{c}{(4\pi Dct)^{3/2}} e^{-\mu_a ct} \left[e^{-\frac{(\rho^2+(z-z_0)^2)}{4Dct}} - e^{-\frac{(\rho^2+(z+z_0)^2)}{4Dct}} \right]$$

A partir de esta expresión se puede calcular la reflectancia difusa sin más que aplicar la ley de Fick:

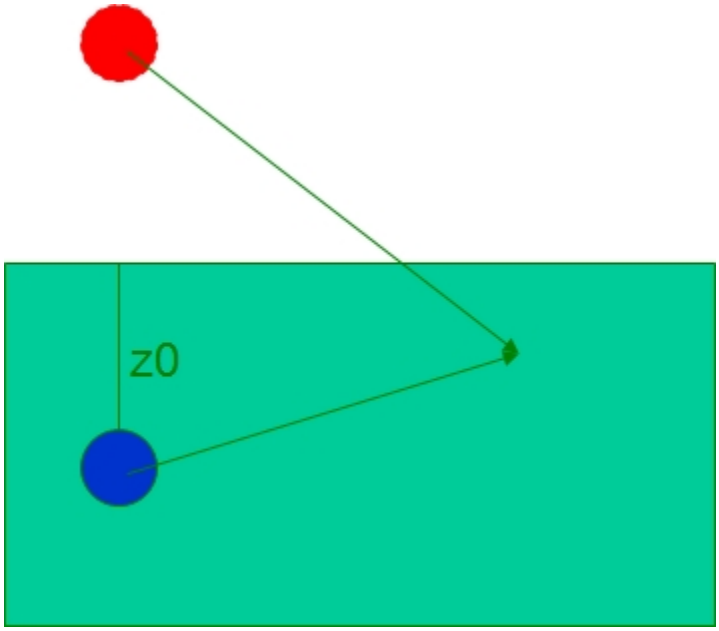
$$R_d(\rho,t) = \frac{c}{(4\pi Dc)^{3/2}} z_0 t^{-5/2} e^{-\mu_a ct} e^{-\frac{(\rho^2+z_0^2)}{4Dct}}$$

Si consideramos el caso en el que se cumple que $\rho >> z$, y tomamos la derivada del ln(R)se llega a que

$$\frac{dR_d(\rho,t)}{d} = \frac{5}{2t} - \mu_a c + \frac{\rho^2}{4Dct}$$

Para tiempos grandes esta expresión tiende a

$$\frac{dR_d(\rho,t)}{d} = -\mu_a c$$



que nos indica que podemos obtener el coeficiente de absorción de un tejido a partir de la pendiente de la curva que da la reflectancia difusa frente al tiempo.

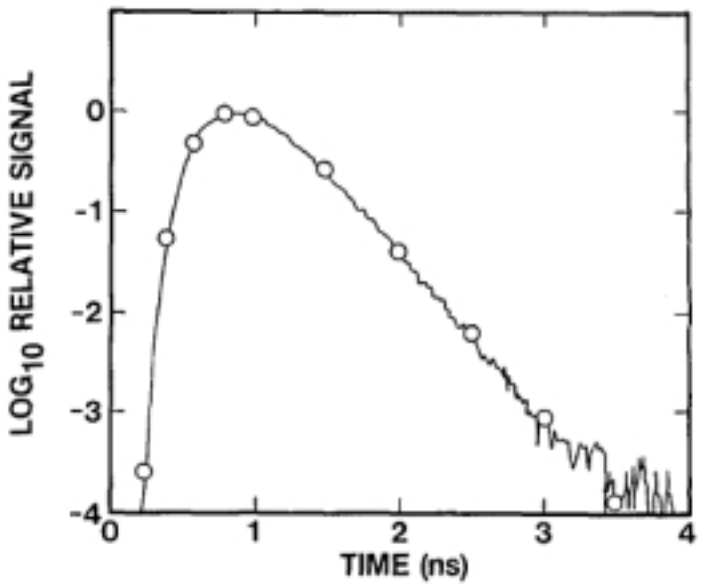
Por otra parte, el coeficiente de scattering reducido $\mu_s' = (1-g)\mu_s$ tambiénn se puede obtener a partir del máxximo de la curva dada por $ln(R(p,t)]$ versus t. En efecto el tiempo en el que se se alcanza el máximo, t_{max} se obtiene haciendo nula la derivada de la función. Ello da lugar a la siguiente condición:

$$\mu_s' = \frac{1}{3\rho^2}\left(4\mu_a c^2 t_{max} + 10 c t_{max}\right) - \mu_a$$

Ejercicio 6

(a) Representar la función F(r,t)=ln (R_d) en el intervalo temporal [0, 4] nanosegundos, teniendo en cuenta que el índice de refracción del tejido muscular es n=1.4 y que $\mu_a = 0.023 mm^{-1}$ y $\mu_s' = 0.85\ mm^{-1}$

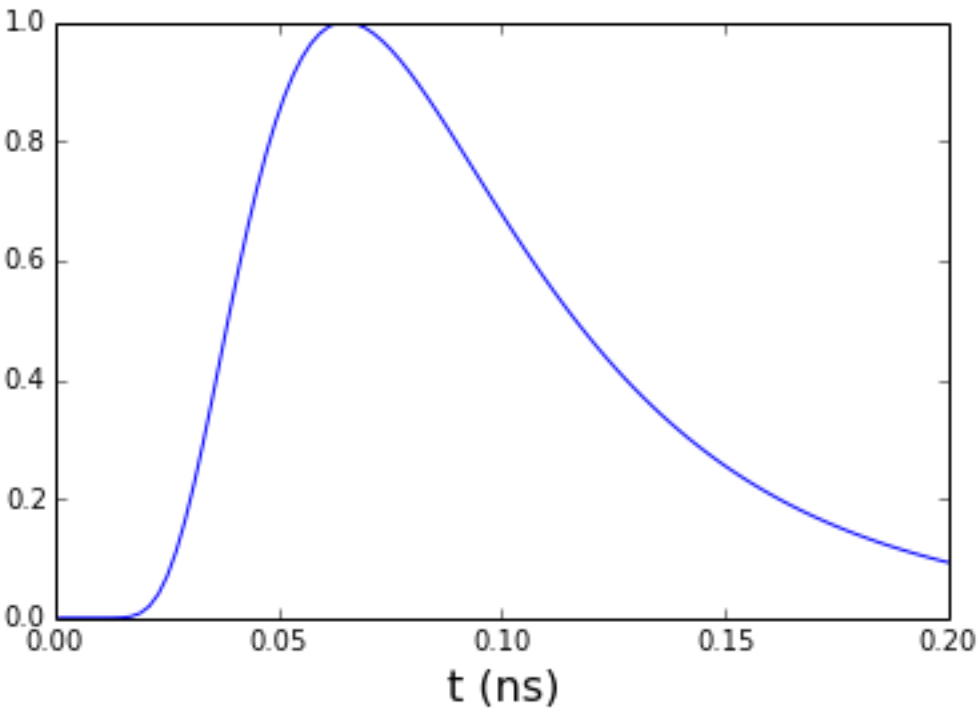
(b) En la figura se representa el logaritmo decimal de la reflectancia medida a través de una muestra de tejido muscular en función del tiempo cuando se ilumina con un pulso laser de 6 ns en $\lambda = 760$ nm. Obtener el coeficiente de absorción del tejido muscular a partir de los datos de la gráfica adjunta.



Solución:

1. Realizamos un pequeño programa que calcula la transmitancia difusa temporal.

```
In [49]: n=1.4;
         mua=0.23; #cm-1
         mus=8.5; #cm-1
         d=1; #cm
         rho=0.4; #cm
         z0 = 1/(mua + mus);
         D = z0/3;
         c=3e10/n
         t=linspace(1,200,100)*1e-12;
         den=4*D*c;

         T=1/((4*pi*D*c)**(1/2))* exp(-mua*c*t)*t**(-5/2)*( (d-z0)*exp(-(d- z0)**2/(den*t))
         - (d+z0)*exp(-(d+ z0)**2/(den*t))  + (3*d-z0)*exp(-(3*d- z0)**2/(den*t))
         - (3*d+z0)*exp(-(3*d+ z0)**2./(den*t)) );
         N=max(T);
         plot(t*1e9,T/N);
         xlabel('t (ns)',fontsize=16);
```



1. Según hemos visto más arriba, las propiedades ópticas del músculo se pueden obtener a partir del valor de la pendiente de la curva, y del máximo de la misma:

En este caso, la pendiente en la zona de tiempos largos es

$$p = \frac{-3 + 0.5}{3 - 1.5} ln(10) = -1.66 x ln(10)\ ns^{-1}$$

Teniendo en cuenta que para tiempos largos se cumple que

$$\frac{dR_d(\rho, t)}{d} = -\mu_a c$$

se tiene que

$$\mu_a = \frac{p}{c} = \frac{-1.66 x ln(10)}{3 x 10^8} = 12.7 m^{-1} = 0.013\ mm^{-1}$$

Por otra parte, podemos calcular el coeficiente de scattering a partir del máximo de la curva. En efecto, en el máximo se tiene que

$$\frac{dR_d(\rho, t)}{d} = \frac{5}{2t} - \mu_a c + \frac{\rho^2}{4Dct}$$

, es decir

$$\rho^2 - 4c^2 D \mu_a t_{max}^2 - 10 c D t_{max} = 0$$

Como

$$D = \frac{1}{\mu_a + \mu_s'}$$

, sustituyendo arriba, se tiene

$$\mu_s' = \frac{1}{3\rho^2}\left[4c^2\mu_a t_{max}^2 + 10cDt_{max}\right] - \mu_a$$

Sustituyendo los valores se llega a

$$\mu_s' = \frac{1}{3 \times (40)^2}\left[4 \times (2.14 \times (10^{11})^2 \times 0.0176 \times (1 \times 10^{-9})^2 + 10 \times (2.14 \times 10^{11} \times (1 \times 10^{-9}\right] - 0.0176 = 1.1\ mm^{-1}$$