

# Prueba 2 - Métodos Numéricos - MEC301

Fecha de entrega: 19 de octubre, 2022

Nombre:

Instrucciones.

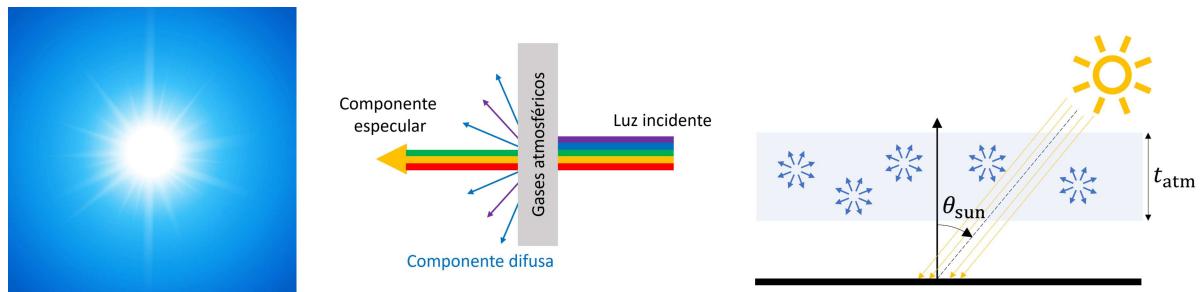
- Complete este archivo con sus respuestas.
- Una vez completado, guarde el archivo como: "prueba2\_apellido\_nombre.ipynb".
- Adjunte el archivo a la plataforma webcurso (solo el archivo \*ipynb)

Definiciones:

- **ndarray**, arreglo `numpy` de dimensión variable
- **callable** variable tipo función.
- **float** variable decimal de tamaño unidimensional

## Problema 1 (3.0 puntos)

El color celeste del cielo está asociado al fenómeno de **scattering de Rayleigh**, mediante el cual la luz del sol es dispersada por los gases en la atmósfera. Producto del *scattering*, la luz solar se divide en una componente especular y una componente difusa. El color celeste del cielo está asociado a la componente difusa.



La **componente difusa de la irradianciapectral del sol**,  $I_{\text{sun,dif}}$ , en función de la longitud de onda  $\lambda$  y el ángulo del sol  $\theta_{\text{sun}}$ , está dada por:

$$I_{\text{sun,dif}}(\lambda, \theta_{\text{sun}}) = \left[ 1 - \exp\left( -f_v \kappa_{\text{sca}} \frac{t_{\text{atm}}}{\cos \theta_{\text{sun}}} \right) \right] I_{\text{sun}}(\lambda) \quad \frac{\text{W}}{\text{m}^2 \cdot \mu\text{m}} \quad (1)$$

donde  $f_v = 1 \times 10^{-8}$  es la fracción de volumen ocupado por las partículas,  $t_{\text{atm}} = 1 \times 10^{11} \mu\text{m}$  es el espesor de la atmósfera,  $I_{\text{sun}}$  es la componente total de la irradianciapectral del sol para  $\theta_{\text{sun}} = 0^\circ$ , y

$$\kappa_{\text{sca}} = \frac{4.4941 \times 10^{-5}}{\lambda^4}, \quad \mu\text{m}^{-1}$$

es el coeficiente de scattering con  $\lambda$  en unidades de  $\mu\text{m}$ .

**En este problema simularemos como el color del cielo cambia con la posición del sol. Utilizaremos el procedimiento de la Comisión internacional de la iluminación (CIE) para convertir el espectro de  $I_{\text{sun,dif}}(\lambda, \theta_{\text{sun}})$  a un código de colores RGB, para un rango de valores de  $\theta_{\text{sun}}$ . El procedimiento será detallado en los pasos (a), (b), (c) y (d).**

**(a) (0.7 pts)** Complete la función `Isun_dif(lam, theta)` para determinar la componente difusa de la radiancia espectral del sol a partir de la ecuación (1).

En la función:

- `lam` corresponde a  $\lambda$  en formato de **ndarray**, en unidades de micrometros ( $\mu\text{m}$ ).
- `theta` corresponde a  $\theta_{\text{sun}}$  en formato de **float**, en unidades de grados ( $^{\circ}$ ).
- El parámetro de salida (return) debe ser una variable **ndarray** con los valores de  $I_{\text{sun,dif}}$  asociados al arreglo `lam`. -Para  $I_{\text{sun}}(\lambda)$  utilice la función `Isun(lam)` del script `blackbody`

Compruebe su función con `Isun_dif(lam=0.3, theta=30) = 12142209.070544254`, y luego grafique `Isun_dif(lam,theta=30)/Isun(lam)` vs `lam`, para  $\lambda \in [0.3, 1.0] \mu\text{m}$  considerando 100 puntos igualmente espaciados. En su gráfico considere:

- Etiqueta '`Longitud de onda, $\lambda$ ($\mu m$)`' en el eje x
- Etiqueta '`Transmitancia difusa`' en el eje y

In [ ]: `from extra_functions import Isun`

```
# función Isun_dif
def Isun_dif(lam,theta):
    # convertimos theta de grados a radianes

    return # ecuación (1)
```

In [ ]: `# Comprobar solución`

In [ ]: `# Gráfico: Isun_dif(Lam,theta)/Isun(Lam) vs Lam`

**(b) (0.8 pts)** Las funciones  $x(\lambda)$ ,  $y(\lambda)$  y  $z(\lambda)$ , representan la sensibilidad de los conos oculares al color rojo, verde y azul, respectivamente. En la siguiente celda, estas funciones están tabuladas a través de las variables `x_cie`, `y_cie`, `z_cie`, respectivamente. Cada valor de este arreglo está asociado a la longitud de onda `lam_cie`, en  $\mu\text{m}$ .

In [1]: `import numpy as np
import matplotlib.pyplot as plt`

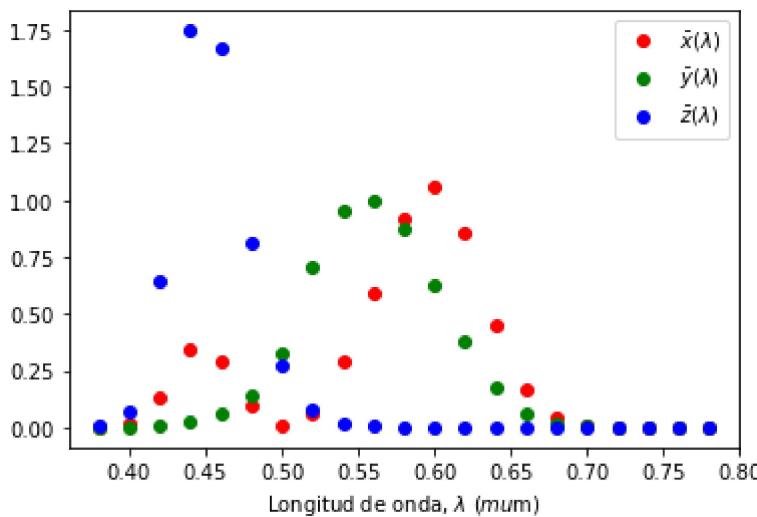
```
# datos tabulados
lam_cie = np.array([0.38, 0.40, 0.42, 0.44, 0.46, 0.48, 0.50, 0.52, 0.54, 0.56, 0.58,
x_cie   = np.array([1.4000e-03, 1.4300e-02, 1.3440e-01, 3.4830e-01, 2.9080e-01, 9.5600e-01,
y_cie   = np.array([0.00e+00, 4.00e-04, 4.00e-03, 2.30e-02, 6.00e-02, 1.39e-01, 3.23e-01,
z_cie   = np.array([6.5000e-03, 6.7900e-02, 6.4560e-01, 1.7471e+00, 1.6692e+00, 8.1300e-01])

# gráfico de Los datos tabulados
plt.plot(lam_cie,x_cie,'o r', label=r'$\bar{x}(\lambda)$')
```

```

plt.plot(lam_cie,y_cie,'o g', label=r'$\bar{x}(\lambda)$')
plt.plot(lam_cie,z_cie,'o b', label=r'$\bar{y}(\lambda)$')
plt.xlabel('Longitud de onda, $\lambda$ ($\mu\text{m}$)')
plt.legend()
plt.show()

```



A partir de los datos tabulados, desarrolle tres funciones `x_cie_fix(lam)`, `y_cie_fix(lam)`, `z_cie_fix(lam)` para determinar el valor de  $\bar{x}(\lambda)$ ,  $\bar{y}(\lambda)$  y  $\bar{z}(\lambda)$ , donde `lam` es la longitud de onda en  $\mu\text{m}$ . Mediante un gráfico, con  $\lambda \in [0.3, 2.0] \mu\text{m}$ , verifique que sus funciones se comporten según lo esperado (es decir, sin divergencia).

In [ ]: # Funciones `x_cie_fix(lam)`, `y_cie_fix(lam)` y `z_cie_fix(lam)`

In [ ]: # Gráfico de `x_cie_fix(lam)`, `y_cie_fix(lam)` y `z_cie_fix(lam)`, para `lam` entre 0.3 y 2.0

(c) (1.0 pts) Desarrolle una función para convertir un espectro de irradiancia  $I_{\text{spec}}(\lambda)$ , en un código de colores  $xyz$  donde:

$$x = \frac{X}{X + Y + Z}, \quad y = \frac{Y}{X + Y + Z}, \quad z = \frac{Z}{X + Y + Z},$$

y  $XYZ$  son los colores de triple estímulo, dados por:

$$X = \int_{0.01}^2 I_{\text{spec}}(\lambda) \bar{x}(\lambda) d\lambda, \quad Y = \int_{0.01}^2 I_{\text{spec}}(\lambda) \bar{y}(\lambda) d\lambda, \quad Z = \int_{0.01}^2 I_{\text{spec}}(\lambda) \bar{z}(\lambda) d\lambda,$$

Considere dos versiones de su función:

1. `spectrum_to_xyz_quad(Ispec)` que calcule las integrales mediante `quad`
2. `spectrum_to_xyz_trapz(Ispec)` que calcule las integrales mediante `trapezoid`. Para los subintervalos, considere  $\lambda \in [0.01, 2] \mu\text{m}$  con una subdivisión de 1000 puntos igualmente espaciados

En ambas funciones,

- el parámetro de entrada `Ispec` es del tipo **callable** y representa a  $I_{\text{spec}}(\lambda)$
- el parámetro de salida debe ser un **ndarray** de tamaño (3,) con los valores de  $xyz$

Como validación, considere `spectrum_to_xyz_quad(Isun) = array([0.32678688, 0.33639789, 0.33681523])`. (similar para `spectrum_to_xyz_trapz(Isun)` ).

```
In [ ]: # (1) versión spectrum_to_xyz_quad  
# validación
```

```
In [ ]: # (2) versión spectrum_to_xyz_trapz  
# validación
```

Finalmente, comente cuál de las dos alternativas es más conveniente.

**Respuesta:**

**(c)** Crear una función `xyz_to_rgb(xyz)` para convertir un código *xyz* a un código *rgb*, mediante:

$$r = \frac{R}{\max(R, G, B)}, \quad g = \frac{G}{\max(R, G, B)}, \quad b = \frac{B}{\max(R, G, B)}$$

donde *R*, *G* y *B* se determinan mediante la solución de:

$$\begin{bmatrix} 0.187 & 0.063 & 0.064 \\ 0.092 & 0.212 & 0.025 \\ 0 & 0.024 & 0.334 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

En la función:

- `xyz` es una variable **ndarray** de tamaño (3,) con los valores de *xyz*
- El parámetro de salida debe ser un **ndarray** de tamaño (3,) con los valores de *rgb*

Como validación considere `xyz_to_rgb(np.array([0.3, 0.3, 0.4])) = array([0.81005352, 0.77822204, 1.])`

```
In [ ]: # Función xyz_to_rgb(xyz)  
# Comprobamos La solución
```

**(d) (0.5 pts)** Complete el siguiente código para determinar el cambio de color del cielo respecto al ángulo del sol  $\theta_{\text{sun}}$ , donde  $\theta_{\text{sun}} \in [0, 89.99]^\circ$  está definido por el arreglo `theta_sun`.

```
In [ ]: theta_sun = np.linspace(0,89.99,100) # rango de ángulo theta del sol  
  
fig, ax = plt.subplots()  
for theta in theta_sun:  
  
    # Función de irradiancia difusa para el ángulo theta  
    # Convertir espectro de irradiancia a xyz  
    xyz = # Convertir xyz a rgb (ASIGNAR RESULTADO A UNA VARIABLE rgb)  
    ax.axvline(theta, color=rgb, linewidth=5) # graficar una Línea de color rgb
```

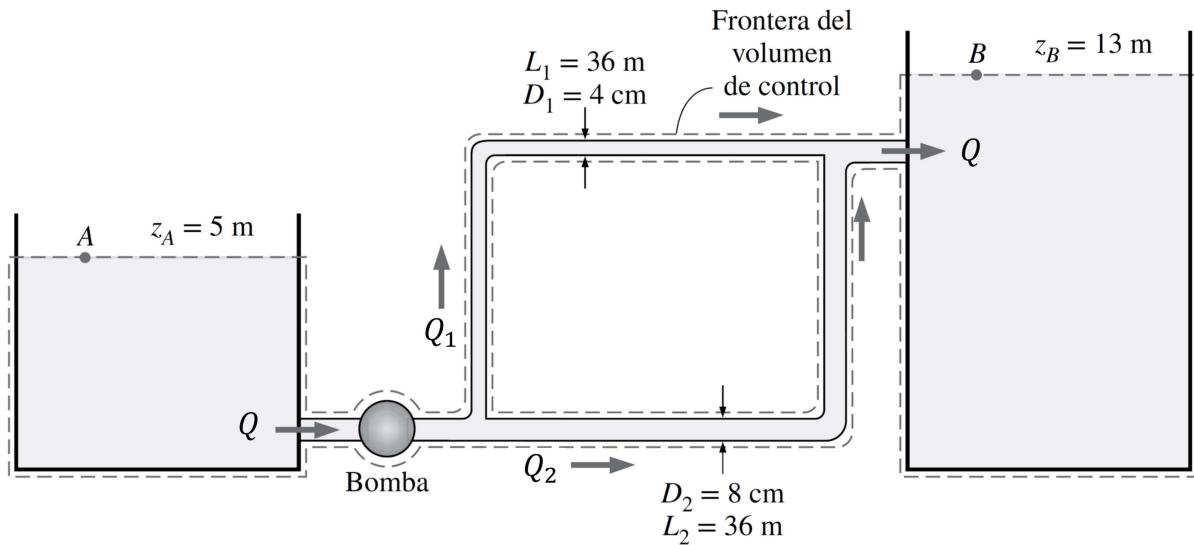
```

ax.set_xlim(0,1)
ax.set_ylim(0,90)
ax.set_xlabel(r'Posición del sol, $\theta_{\mathrm{sun}}$ (°)')
ax.set_ylabel('')
ax.set_yticks([])
plt.show()

```

## Problema 2 (3.0 puntos)

Se utiliza un sistema de bombeo para mover agua desde el estanque A al estanque B, como se indica la figura.



El agua tiene una densidad,  $\rho = 1000 \text{ kg/m}^3$ , y viscosidad cinemática,  $\nu = 1 \times 10^{-6} \text{ m}^2/\text{s}$ . La rugosidad de la tubería  $\varepsilon = 0.045 \text{ mm}$ .

Para un caudal  $Q$ , la carga hidrostática del sistema,  $H_{\text{sys}}$ , y el caudal en cada línea  $Q_1$  y  $Q_2$  están definidos por el siguiente sistema de ecuaciones:

$$H_{\text{sys}} = z_B - z_A + h_L \quad (2.a)$$

$$h_L = h_{L,1} = h_{L,2} \quad (2.b)$$

$$Q_1 + Q_2 = Q \quad (2.c)$$

El parámetro  $h_{L,i}$  ( $i = 1, 2$ ) representa las pérdidas mayores:

$$h_{L,i} = f_i \frac{L_i}{D_i} \frac{V_i^2}{2g} \quad (3)$$

donde  $f_i$ ,  $L_i$ ,  $D_i$  y  $V_i$  representan, respectivamente, el factor de fricción (adimensional), la longitud (m), el diámetro (m) y la velocidad (m/s) en la línea  $i$ . La aceleración de gravedad  $g = 9.81 \text{ m/s}^2$ .

**(a) (0.6 pts)** Complete la función `get_hL(x,D,L)` para determinar  $h_L$  en m, usando la ecuación (3), donde:

- `x` : caudal en  $\text{m}^3/\text{s}$  (formato **float**)

- D : Diametro en  $m$  (formato **float**)
- L : Largo de la tubería en  $m/s$  (formato **float**)

Como guía, considere los siguientes pasos dentro de su función:

- Paso 1: Definición de los parámetros  $g$ ,  $\varepsilon$ ,  $\rho$  y  $\nu$ .
- Paso 2: Cálculo de la velocidad del flujo en el ducto, mediante la expresión:  

$$V = \frac{Q}{\pi D^2/4}$$
- Paso 3: Cálculo del número de Reynolds  $Re = \frac{VD}{\nu}$ , y rugosidad relativa  $\varepsilon_r = \varepsilon/D$ .
- Paso 4: Cálculo del factor de fricción  $f$ , mediante la función `f_colebrook` del script `extra_functions`.
- Paso 5: Cálculo de pérdidas mayores de presión  $h_L$ .

Como validación, considere `get_hL(0.00415, 0.04, 36) = array([11.04823106])`

```
In [ ]: from extra_functions import f_colebrook

# Función get_hL
def get_hL(x, D, L):

    # paso 1: Definición de constantes y propiedades

    # paso 2: Cálculo de la velocidad del flujo

    # paso 3: Número de Reynolds y rugosidad relativa

    # paso 4: Cálculo del factor de fricción a partir de Re y eps_r

    # paso 5. Cálculo de perdidas mayores de presión

    return hL

# validación
```

**(b) (1.0 pts)** Desarrolle una función `pump_system_root(Q)` tal que, para un caudal  $Q$  en  $m^3/h$ , determine el valor de  $Q_1$  (en  $m^3/s$ ),  $Q_2$  (en  $m^3/s$ ) y  $H_{sys}$  (en m) que satisface el sistema de ecuaciones (2a), (2b) y (2c).

**NOTA** Considere `get_hL` para determinar  $h_{L,1}$  y  $h_{L,2}$ .

Verifique que `pump_system_root(Q = 108) = (0.004149837300192359, 0.02585016269980764, 19.047396674297296)`, donde el primer valor corresponde a  $Q_1$ , el segundo a  $Q_2$  y el tercero a  $H_{sys}$ .

```
In [ ]: # crear función

# validación
```

**(c) (0.7 pts)** En la siguiente celda se presentan datos tabulados para la curva de rendimiento de una bomba NK100 - 315.

```
In [ ]: # Datos tabulados correspondientes a la curva de rendimiento de una bomba NK100-315
import numpy as np
Qbba = np.array([20.44, 102.22, 179.56, 235.56, 283.56, 320.89, 357.33, 400])
Hbba = np.array([150.64, 149.14, 147.64, 142.70, 134.55, 126.18, 117.38, 104])
```

A partir de estos datos **grafique la curva de demanda del sistema y la curva de rendimiento de la bomba.** En el gráfico considere:

- Curva de demanda del sistema `Hsys` vs `Q`, con `Hsys` obtenido por `pump_system_root(Q)`, donde `Q` es un arreglo de 100 puntos igualmente espaciados en el intervalo  $Q \in [1, 400]$  m<sup>3</sup>/h. Leyenda 'Curva de demanda'
- Curva de rendimiento de la bomba utilizando los datos tabulados `Qbba` y `Hbba`, unidos por línea punteada. Leyenda 'Curva de rendimiento'
- Etiqueta en el eje horizontal 'Caudal, Q (m<sup>3</sup>/h)'
- Etiqueta en el eje vertical 'Altura hidrostática (m)'

```
In [ ]: # Gráfico de curvas de rendimiento y demanda
```

(d) (0.7 pts) Mediante `root_scalar`, con un método de intervalo cerrado, determine:

- $Q_{op}$  en m<sup>3</sup>/h
- $\dot{W}_{bba}$  en kW.

donde  $Q_{op}$  corresponde al caudal dado por el cruce de las curvas de demanda y rendimiento, y la potencia de la bomba,  $\dot{W}_{bba} = \rho g H_{op} Q_{op}$ .

Utilice `print` para mostrar su resultado

```
In [ ]: # respuesta
```