Anexos

Anexo A

Conceptos físico-biológicos en los balances de materia

Los balances de masa son considerados una herramienta muy potente en el análisis de ingeniería. Muchas situaciones complejas se simplifican pudiendo observar el movimiento de la masa y relacionando aquello que entra en un sistema con aquello que lo abandona. Preguntas como: ¿Qué cantidad de oxígeno hay en el biofiltro? ¿Qué fracción de sustrato no se convierte en producto? ¿Qué cantidad hay de gases de escape? pueden ser respondidas con los balances de materia.[17]

Nos centraremos en los principios termodinámicos para poder configurar los balances de materia que regirán nuestro biorreactor de estudio. En primer lugar, debemos definir el concepto de *sistema*, que consiste en cualquier materia identificada para ser investigada, y que se encuentra apartada de su entorno por las fronteras del sistema. Estas fronteras pueden ser físicas y tangibles, como las paredes de un reactor, o pueden ser virtuales. Nuestro sistema, además, será abierto, pues permite el intercambio de materia con el entorno (entran y salen corrientes de distintos sustratos y productos). En la figura A.1 podemos observar gráficamente el concepto de sistema previamente definido.

Por lo tanto, tomando medidas de la masa que entra a un sistema y de aquella que sale del mismo, la diferencia entre las mismas es debida a la generación o consumo de las mismas por reacción o acumulación en el sistema. Un balance de masa para este sistema puede ser escrito, de manera genérica, de la siguiente forma:

$$masa_{acumulada} = masa_{in} - masa_{out} + masa_{gen} - masa_{cons} \tag{A.1}$$

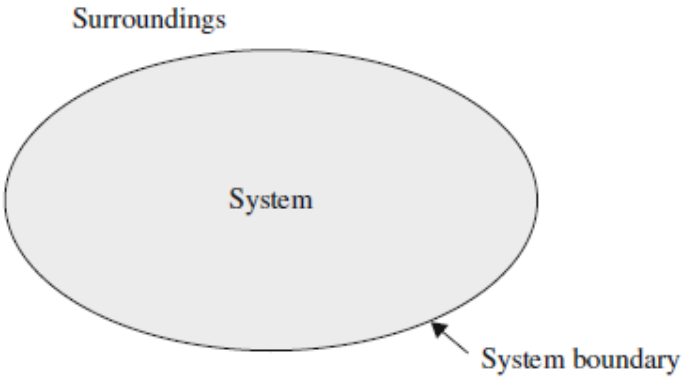

Figura A.1: Esquema del concepto de sistema[17]

Considerando un sistema genérico como el de la figura A.2 en el que tienen lugar las reacciones químicas (en nuestro caso, el biorreactor):

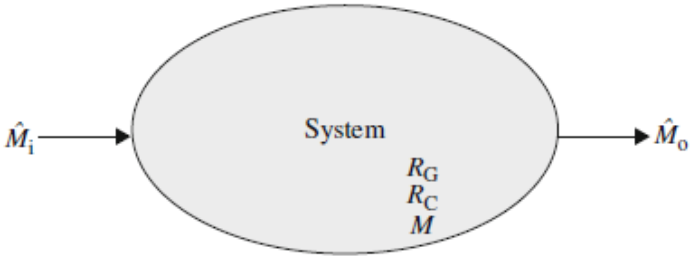

Figura A.2: Esquema de los flujos en un sistema[17]

$\hat{M}_o$ es el flujo másico de salida, $\hat{M}_i$ es el flujo másico de entrada, R_G es la masa generada de la especie, R_C es la masa consumida por la reacción y M es la masa de la especie en el sistema. Si nos centramos en un tiempo infinitesimal Δt lo suficientemente pequeño como para tratar a los flujos y tasas de reacción como constantes, los términos del balance de masa A.1 se pueden expresar de la siguiente forma:

$$\Delta M = \hat{M}_i \Delta t - \hat{M}_o \Delta t + R_G \Delta t - R_c \Delta t \tag{A.2}$$

Dividiendo por Δt tenemos:

$$\frac{\Delta M}{\Delta t} = \hat{M}_i - \hat{M}_o + R_G - R_c \tag{A.3}$$

Cuando Δt tiende a 0 podemos expresar la expresión anterior como:

$$\frac{dM}{dt} = \lim_{\Delta t \to 0} \frac{\Delta M}{\Delta t} = \hat{M}_i - \hat{M}_o + R_G - R_c \tag{A.4}$$

Los balances se harán en términos de concentración, por lo que los flujos másicos irán divididos por el volumen. En términos generales, los balances tendrán la siguiente forma:

$$\frac{dC}{dt} = \frac{C_{entrada}}{V} - \frac{C_{salida}}{V} + \text{generación} - \text{consumo} \tag{A.5}$$

Los dos primeros términos hacen referencia a las tasas de entrada y salida de la especie al volumen de control, que dependen de la concentración a la entrada y a la salida, y del caudal, incluido en el volumen V.

El tercer término representa la cantidad de componente producido dentro del sistema (por ejemplo, por una reacción química o biológica). Por otra parte, el último término hace referencia a la cantidad de componente que se consume dentro del sistema, debido a reacciones u otros procesos.

A.1. Cinética de Michaelis-Menten

El término de generación en el balance de masa de las ecuaciones planteadas se ve caracterizado por la cinética de Michaelis-Menten, que describe la velocidad de reacción de muchas reacciones enzimáticas. Su nombre es en honor a Leonor Michaelis y Maud Menten. Este modelo solo es válido cuando la concentración del sustrato es mayor que la concentración de la enzima, y para condiciones de estado estacionario, o sea que la concentración del complejo enzima-sustrato es constante [35].

La cinética de Michaelis-Menten es igualmente válida para describir la velocidad de consumo de sustrato por parte de los microorganismos adheridos o en suspensión. Igualmente, desarrollaremos su origen matemático en el supuesto de la enzima y sustrato.

A.1.1. Derivación de la ecuación de Michaelis-Menten

La cinética enzimática de Michaelis-Menten se basa en el siguiente mecanismo de reacción:

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} ES \overset{k_2}{\rightarrow} E + P \tag{A.6}$$

donde:

- E es la enzima libre.
- S es el sustrato.
- ES es el complejo enzima-sustrato.
- P es el producto.

Las constantes de velocidad asociadas son:

- k_1: constante de velocidad de formación del complejo ES.
- k_{-1}: constante de velocidad de disociación del complejo ES en enzima y sustrato.
- k_2: constante de velocidad de transformación de ES en enzima y producto.

Se asume que, tras un breve periodo inicial, la concentración del complejo ES permanece prácticamente constante durante la mayor parte de la reacción. Por tanto:

$$\frac{d[ES]}{dt} = 0 \tag{A.7}$$

El balance de velocidades para la formación y desaparición del complejo ES es:

$$\frac{d[ES]}{dt} = k_1[E][S] - (k_{-1} + k_2)[ES] = 0 \tag{A.8}$$

De aquí se despeja la concentración de ES:

$$[ES] = \frac{k_1[E][S]}{k_{-1} + k_2} \tag{A.9}$$

Se define la constante de Michaelis-Menten como:

$$K_m = \frac{k_{-1} + k_2}{k_1} \tag{A.10}$$

Por tanto:

$$[ES] = \frac{[E][S]}{K_m} \tag{A.11}$$

La velocidad de formación de producto es:

$$v = \frac{d[P]}{dt} = k_2[ES] \tag{A.12}$$

Sustituyendo la expresión de $[ES]$:

$$v = k_2 \frac{[E][S]}{K_m} \tag{A.13}$$

Como la concentración total de enzima es:

$$[E]_0 = [E] + [ES] \tag{A.14}$$

Se despeja $[E]$:

$$[E] = [E]_0 - [ES] \tag{A.15}$$

Sustituyendo en la ecuación de v:

$$v = k_2 \frac{([E]_0 - [ES])[S]}{K_m} \tag{A.16}$$

Pero como ya tenemos $v = k_2[ES]$, podemos expresar todo en función de $[ES]$:

$$v = k_2[ES] \tag{A.17}$$

y usando la relación:

$$[ES] = \frac{[E]_0[S]}{K_m + [S]} \tag{A.18}$$

se obtiene la ecuación final:

$$v = \frac{V_{\max}[S]}{K_m + [S]} \tag{A.19}$$

donde:

$$V_{\max} = k_2[E]_0 \tag{A.20}$$

La velocidad V indica el número de reacciones por segundo que son catalizadas. Con concentraciones crecientes de sustrato, S, la enzima va acercándose asintóticamente a su velocidad máxima, V_{max}, pero nunca la alcanza. Entonces, aunque la concentración de sustrato a V_{max} no pueda ser medida exactamente, las enzimas pueden ser caracterizadas por la concentración de sustrato a la cual la velocidad de reacción es la mitad de la velocidad máxima. Esta concentración de sustrato se conoce como constante de Michaelis-Menten (K_M).

A.2. Ley de transferencia de masa

El término **transferencia de masa convectiva** se refiere a la transferencia de masa que ocurre en presencia de movimiento del fluido en masa. La difusión molecular se produce siempre que existe un gradiente de concentración. Sin embargo, si el fluido también se desplaza en bloque, la velocidad global de transferencia de masa será mayor debido a la contribución de las corrientes convectivas.

El análisis de la transferencia de masa es especialmente importante en sistemas multifásicos, donde las capas límite interfaciales representan una resistencia significativa a la transferencia de masa. Se desarrollará una expresión para la velocidad de transferencia de masa aplicable a capas límite de transferencia de masa[17].

La velocidad de transferencia de masa es directamente proporcional al área disponible para la transferencia y a la fuerza impulsora del proceso de transferencia. Esto se puede expresar como:

$$\text{Velocidad de transferencia} \propto \text{Área de transferencia} \times \text{Fuerza impulsora}$$

El coeficiente de proporcionalidad en esta ecuación se denomina **coeficiente de transferencia de masa**, de forma que:

$$\begin{aligned}\text{Velocidad de transferencia} = {} & \text{Coeficiente de transferencia de masa} \\ & \times \text{Área de transferencia} \\ & \times \text{Fuerza impulsora}\end{aligned}$$

Para cada fluido en una frontera de fase, la fuerza impulsora para la transferencia de masa del componente A a través de la capa límite se puede expresar en función de la diferencia de concentración de A a través de la película de fluido. Por lo tanto, la velocidad de transferencia de masa desde el seno del fluido a través de la capa límite hasta la interfaz es:

$$N_A = ka\Delta C_A = ka\left(C_{Ab} - C_{Ai}\right) \tag{A.21}$$

Donde:

- N_A es la velocidad de transferencia de masa del componente A,

- k es el coeficiente de transferencia de masa,
- a es el área disponible para la transferencia de masa,
- C_{Ab} es la concentración de A en el seno del fluido, alejado de la frontera de fase,
- C_{Ai} es la concentración de A en la interfaz.

La ecuación A.21 se suele emplear para representar la velocidad volumétrica de transferencia de masa, por lo que las unidades de N_A son, por ejemplo, gmol $m^{-3}s^{-1}$.

De acuerdo con esta representación, a es el área interfacial por unidad de volumen, con dimensiones L^{-1} y unidades como m^2m^{-3} o m^{-1}. Las dimensiones del coeficiente de transferencia de masa k son L T^{-1}; sus unidades en el SI son m s^{-1}.

La ecuación A.21 indica que la velocidad de transferencia de masa convectiva se puede incrementar aumentando el área disponible para la transferencia de masa, la diferencia de concentración entre el seno del fluido y la interfaz, y el valor del coeficiente de transferencia de masa. Por analogía con ecuaciones de transferencia de calor, la ecuación A.21 puede escribirse como:

$$N_A = \frac{\Delta C_A}{R_m} \tag{A.22}$$

donde R_m es la resistencia a la transferencia de masa:

$$R_m = \frac{1}{ka} \tag{A.23}$$

La transferencia de masa acoplada al flujo de fluido es un proceso más complejo que la transferencia de masa puramente difusiva. El valor del coeficiente de transferencia de masa k refleja la contribución de todos los procesos en el sistema que afectan a la capa límite y depende de los efectos combinados de la velocidad del flujo, la geometría del sistema de transferencia de masa y las propiedades del fluido, tales como la viscosidad y la difusividad.

La transferencia de masa gas-líquido es de gran importancia en los bioprocesos debido a la necesidad de oxígeno en los cultivos celulares aerobios. Analizamos la transferencia de un soluto, como el oxígeno u otros elementos, desde la fase gaseosa a la fase líquida.

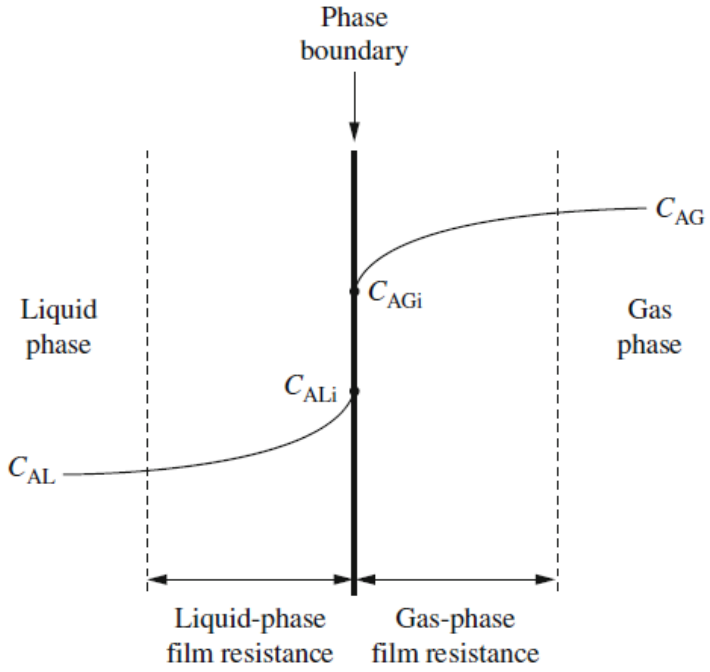

Figura A.3: Gradientes de concentración en la transferencia de masa[17]

La figura A.3 muestra la situación en una interfaz entre fases gaseosa y líquida que contienen al componente A. Supongamos que A se transfiere desde la fase gaseosa hacia la fase líquida. La concentración de A en la fase líquida es C_{AL} en el seno del líquido y C_{ALi} en la interfaz. En la fase gaseosa, la concentración es C_{AG} en el seno del gas y C_{AGi} en la interfaz.

A partir de la ecuación A.21 la velocidad de transferencia de masa de A a través de la capa límite en la fase gaseosa es:

$$N_A^G = k_G a \left(C_{AG} - C_{AGi}\right) \tag{A.24}$$

y la velocidad de transferencia de masa de A a través de la capa límite en la fase líquida es:

$$N_A^L = k_L a \left(C_{ALi} - C_{AL}\right) \tag{A.25}$$

donde k_G y k_L son los coeficientes de transferencia de masa en la fase gaseosa y líquida, respectivamente.

Para eliminar las concentraciones interfaciales C_{AGi} y C_{ALi}, se deben combinar las ecuaciones anteriores. Si se asume que existe equilibrio en la interfaz,

se puede relacionar C_{AGi} y C_{ALi}. Para concentraciones diluidas de la mayoría de los gases (y en un amplio rango para algunos), la concentración en equilibrio en la fase gaseosa es una función lineal de la concentración en la fase líquida. Así:

$$C_{AGi} = mC_{ALi} \tag{A.26}$$

o equivalentemente:

$$C_{ALi} = \frac{C_{AGi}}{m} \tag{A.27}$$

donde m es el coeficiente de distribución. Estas relaciones de equilibrio se pueden incorporar a las ecuaciones A.24 y A.25 en estado estacionario.

$$N_A = \left(\frac{1}{k_G a} + \frac{1}{m k_L a}\right)^{-1} (C_{AG} - mC_{AL}) \tag{A.28}$$

o bien:

$$N_A = \left(\frac{1}{m k_G a} + \frac{1}{k_L a}\right)^{-1} \left(\frac{C_{AG}}{m} - C_{AL}\right) \tag{A.29}$$

Se definen los coeficientes de transferencia de masa globales a partir de las ecuaciones anteriores. El coeficiente global respecto a la fase gaseosa K_G es:

$$\frac{1}{K_G a} = \frac{1}{k_G a} + \frac{1}{m k_L a} \tag{A.30}$$

y el coeficiente global respecto a la fase líquida K_L es:

$$\frac{1}{K_L a} = \frac{1}{m k_G a} + \frac{1}{k_L a} \tag{A.31}$$

Por tanto, la velocidad de transferencia de masa en sistemas gas-líquido puede expresarse como:

$$N_A = K_G a \, (C_{AG} - mC_{AL}) \tag{A.32}$$

o

$$N_A = K_L a \left(\frac{C_{AG}}{m} - C_{AL}\right) \tag{A.33}$$

Estas ecuaciones se expresan comúnmente en función de las concentraciones en equilibrio. Así, mC_{AL} es igual a C^*_{AG}, la concentración del componente A en fase gaseosa en equilibrio con C_{AL}, y C_{AG}/m es igual a C^*_{AL}, la concentración en fase líquida en equilibrio con C_{AG}. Entonces:

$$N_A = K_G a \left(C_{AG} - C^*_{AG}\right) \tag{A.34}$$

$$N_A = K_L a \left(C^*_{AL} - C_{AL}\right) \tag{A.35}$$

Estas expresiones pueden simplificarse si la resistencia al transporte de masa se concentra en una sola de las fases. Si el soluto A es muy soluble en el líquido (por ejemplo, el amoníaco en agua), la resistencia del lado líquido es despreciable, lo que implica que $k_L a$ es muy grande y:

$$K_G a \approx k_G a \tag{A.36}$$

entonces la ecuación se reduce a:

$$N_A = k_G a \left(C_{AG} - C^*_{AG}\right) \tag{A.37}$$

En cambio, si el soluto es poco soluble en el líquido (como el oxígeno en agua), la resistencia dominante es del lado líquido. En este caso:

$$K_L a \approx k_L a \tag{A.38}$$

y la ecuación se simplifica a:

$$N_A = k_L a \left(C^*_{AL} - C_{AL}\right) \tag{A.39}$$

A.2.1. Ley de Henry

La Ley de Henry fue formulada en 1803 por William Henry. Enuncia que a una temperatura constante, la cantidad de gas disuelta en un líquido es directamente proporcional a la presión parcial que ejerce ese gas sobre el líquido [36]. Matemáticamente se formula del siguiente modo:

$$C_{gas} = k \cdot P_{gas} \tag{A.40}$$

Donde:

- P_{gas} es la presión parcial del gas.
- C_{gas} es la concentración del gas.
- k es la constante de Henry, que depende de la naturaleza del gas, la temperatura y el líquido.

Un cambio de presión no influye de forma apreciable en la solubilidad de sólidos en líquidos o de líquidos en líquidos. Sin embargo, la de los gases en los disolventes aumenta cuando se incrementa la presión parcial de los gases. La solubilidad de un gas depende de la presión y la temperatura. Cuando éste se encuentra encerrado en una disolución saturada, existe un equilibrio entre el gas y la disolución [37].

Si la presión aumenta, el equilibrio se moverá en la dirección en la cual se reduce la presión —principio de Le Chatelier—. La presión se puede reducir si hay más gas disuelto en el disolvente. Así, la solubilidad o concentración de un gas, en un disolvente dado, incrementa con el aumento de la presión.

A.3. Ley de Fick de difusión

La primera ley de Fick describe cómo el flujo de partículas se relaciona con el gradiente de concentración del mismo [38]. La ley establece que el flujo es proporcional al gradiente de concentración a través de la fórmula siguiente:

$$J = -D\frac{dC}{dx} \tag{A.41}$$

- J es el flujo de difusión (cantidad de sustancia que pasa por unidad de área por unidad de tiempo). Se mide en $\mathrm{g}/(cm^2 \cdot h)$.
- D es el coeficiente de difusión. Se mide en cm^2/s.
- $\frac{dC}{dx}$ es el gradiente de concentración. Se mide en $\mathrm{g}/cm \cdot h$.

El signo negativo indica que el flujo circula de regiones de alta concentración a regiones de baja concentración.

Cuando trabajamos en sistemas reales como el biorreactor de estudio, se adapta la Ley de Fick para una geometría y condiciones específicas [18].

Se tiene:

- Un medio poroso de porosidad ϕ
- Un área de difusión efectiva a (superficie de transferencia por unidad de volumen), que se mide en cm^2/cm^3.
- Un espesor de capa de difusión δ_L, que se mide en cm.

- Concentraciones en dos puntos: C_L (en el líquido) y C_B (en el biofilm), que se miden en g/h.

Se hace la suposición de gradiente lineal (condición estacionaria):

$$\frac{dC}{dx} \approx \frac{C_L - C_B}{\delta_L} \tag{A.42}$$

Y, por lo tanto, la Ley de Fick se expresa como:

$$J = -D\frac{C_L - C_B}{\delta_L} \tag{A.43}$$

En un medio poroso, el coeficiente de difusión se ajusta, pero también se tiene en cuenta que no todo el área es efectiva para la transferencia, y que solo una fracción ϕ del volumen es fluido disponible para difundir.

Con esto, el flujo de masa total disponible por volumen de reactor se expresa como:

$$J = \frac{a \cdot D}{\phi \cdot \delta_L}(C_L - C_B) \tag{A.44}$$

Anexo B

Variación axial de concentraciones. Implementación en BioSteam

B.1. Librería BioSteam

BioSTEAM (*Biorefinery Simulation and Techno-Economic Analysis Modules*) es una herramienta de código abierto desarrollada en Python para la simulación de procesos bioquímicos y biotecnológicos, con especial aplicación en el ámbito de las biorrefinerías [39]. Esta librería permite modelar, de forma flexible y modular, operaciones unitarias, balances de materia y energía, así como realizar estudios tecno-económicos preliminares de procesos productivos a distintas escalas.

En el contexto de este Trabajo Fin de Grado, se plantea la utilización de BioSTEAM como soporte para la simulación de un biorreactor tubular (PFR, *Plug Flow Reactor*). Debido a que BioSTEAM no cuenta de forma nativa con una unidad de proceso tipo PFR, se recurre a una metodología habitual en ingeniería química y bioprocesos: modelar el comportamiento de un reactor PFR como una serie de reactores de tanque agitado continuo (CSTR) conectados en serie. Esta aproximación permite aproximar numéricamente el perfil de concentración y conversión a lo largo del reactor, simulando de forma razonable el comportamiento de un flujo-pistón.

Mediante la definición de los componentes implicados, las corrientes de entrada y salida, y las unidades de reacción configuradas como CSTR, es posible simular las condiciones de operación del biorreactor y evaluar:

- Los balances de materia y energía para cada etapa.
- La evolución de la concentración de los reactivos y productos a lo largo del reactor.
- El tiempo de residencia global y su relación con la conversión obtenida.
- Diferentes escenarios de operación, modificando parámetros como el caudal de alimentación, la carga de biomasa o el volumen de los reactores intermedios.

De esta manera, BioSTEAM se presenta como una herramienta de gran utilidad para complementar el diseño y análisis del biorreactor propuesto en este TFG, aportando valor tanto desde el punto de vista técnico como económico.

B.2. Implementación del código

A continuación, se implementa el código utilizado en BioSteam para calcular la variación de concentraciones de cada elemento a lo largo de la longitud del biorreactor.

```
import biosteam as bst
from biosteam import Stream, System, Unit
import thermosteam as tmo
from thermosteam import Chemicals
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

# Definir las sustancias quimicas que intervienen en el biorreactor

chemicals = Chemicals(['NH3', 'NaHCO3', 'NaNO3', 'O2'])
bst.settings.set_thermo(chemicals)

# Crear las corrientes, 4 de entrada y 4 de salida

NH3_in = tmo.Stream(ID='NH3', NH3=3.6, phase='g', units='g/hr')
NaHCO3_in = tmo.Stream(ID='NaHCO3', NaHCO3=1.65, phase='l', units='g/hr')
O2_in = tmo.Stream(ID='O2', O2=5.18, phase='g', units='g/hr')
NaNO3_in = tmo.Stream(ID='NaNO3')
NH3_out = tmo.Stream(ID='NH3')
NaHCO3_out = tmo.Stream(ID='NaHCO3')
O2_out = tmo.Stream(ID='O2')
NaNO3_out = tmo.Stream(ID='NaNO3')

# Creamos una clase personalizada para el biorreactor

class biofilter(bst.Unit):

    _N_ins = 4 # Numero de entradas
    _N_outs = 4 # Numero de salidas
```

```
'Funci n␣de␣inicializacion'

def __init__(self, ID, ins, outs, V, C_in_NH3_gas, C_in_O2_gas,
    C_in_NH3_liq, C_in_NaHCO3, C_in_O2_liq, C_in_NaNO3,
k_La_NH3, k_H_NH3, P_NH3, k_La_O2, k_H_O2, P_O2, v_max, K_m_1, K_m_2,
    delta_L, a, D_NH3, D_O2, D_NaHCO3, D_NaNO3,  phi):
    super().__init__(ID, ins, outs)
    self.V = V                          # Volumen del reactor [L]
    self.C_in_NH3_gas = C_in_NH3_gas    # Concentracion a la entrada de
        NH3 [g/hr]
    self.C_in_O2_gas = C_in_O2_gas      # Concentracion a la entrada de
        NH3 [g/hr]
    self.C_in_NH3_liq = C_in_NH3_liq    # Concentracion a la entrada de
        NH3 [g/hr]
    self.C_in_NaHCO3 = C_in_NaHCO3      # Concentracion a la entrada de
        NaHCO3 [g/hr]
    self.C_in_O2_liq = C_in_O2_liq      # Concentracion a la entrada de
        O2 [g/hr]
    self.C_in_NaNO3 = C_in_NaNO3        # Concentracion a la entrada de
        NaNO3 [g/hr]
    self.k_La_NH3 = k_La_NH3            # Coeficiente de transferencia
        de masa [1/h]
    self.k_H_NH3 = k_H_NH3              # Constante de Henry [g/L/atm]
    self.P_NH3 = P_NH3                  # Presion parcial de NH3 [atm]
    self.k_La_O2 = k_La_O2              # Coeficiente de transferencia
        de masa [1/h]
    self.k_H_O2 = k_H_O2                # Constante de Henry [g/L/atm]
    self.P_O2 = P_O2                    # Presion parcial de O2 [atm]
    self.v_max = v_max                  # Tasa maxima de reaccion [g/L/h
        ]
    self.K_m_1 = K_m_1                  # Constante de Michaelis-Menten
        para el sustrato 1 [g/L]
    self.K_m_2 = K_m_2                  # Constante de Michaelis-Menten
        para el sustrato 2 [g/L]
    self.delta_L = delta_L              # Grosor efectivo de la
        biopelicula [cm]
    self.a = a                          # Area especifica [cm2]
    self.D_NH3 = D_NH3                  # Coeficiente de difusion de NH3
         en el liquido [cm2/s]
    self.D_O2 = D_O2                    # Coeficiente de difusion de O2
        en el liquido [cm2/s]
    self.D_NaHCO3 = D_NaHCO3            # Coeficiente de difusion de
        NaHCO3 [cm2/s]
    self.D_NaNO3 = D_NaNO3              # Coeficiente de difusion de
        NaNO3 [cm2/s]
    self.phi = phi                      # Porosidad del medio
    self.solution = None                # Aqui almacenaremos la solucion
         del ODE

def _run(self):

    'Asociamos␣cada␣especie␣quimica␣a␣una␣entrada␣del␣biorreactor␣y␣una␣
        salida'

    NH3_in = self.ins[0]
    NaHCO3_in = self.ins[1]
    O2_in = self.ins[2]
    NaNO3_in = self.ins[3]
    NH3_out = self.outs[0]
    NaHCO3_out = self.outs[1]
    O2_out = self.outs[2]
```

```
        NaNO3_out = self.outs[3]

        'Usamos␣la␣ley␣de␣Henry␣para␣calcular␣la␣concentracion␣de␣amoniaco␣y
            ␣de␣oxigeno␣en␣la␣interfase'

        C_interfase_NH3 = self.k_H_NH3 * self.P_NH3
        C_interfase_O2 = self.k_H_O2 *  self.P_O2

        'Constantes␣de␣eficiencia␣y␣coeficientes␣estequiometricos␣que␣
            intervendran␣en␣las␣ODEs'
        eficiencia = 0.99
        coef_esteq_NaHCO3 = 0.098
        coef_esteq_NaNO3 = 0.98
        coef_esteq_O2 = 1.863

        'Calculo␣de␣la␣tasa␣de␣reaccion␣basado␣en␣la␣cinetica␣de␣Michaelis-
            Menten'

        def tasa_reaccion(C1, C2):
            return self.v_max * (C1 / (self.K_m_1 + C1)) * (C2 / (
                coef_esteq_O2 * (self.K_m_2 + C2)))

        'Definicion␣del␣sistema␣de␣ecuaciones␣diferenciales'

        def balance_masa(y,t):
            # Variables de estado
            C_NH3_gas, C_O2_gas, C_NH3_liq, C_NaHCO3, C_O2_liq, C_NaNO3,
                C_NH3_biofilm, C_NaHCO3_biofilm, C_O2_biofilm,
                C_NaNO3_biofilm, C_biofilm = y

            # Tasas de reaccion para el liquido y el biofilm
            mu_reaccion = tasa_reaccion(C_NH3_liq, C_O2_liq)
            mu_biofilm = tasa_reaccion(C_biofilm, C_O2_gas)

            # Termino de difusion del liquido al biofilm basado en la Ley de
                 Fick de difusion
            difusion_NH3 = (self.a * self.D_NH3 / (self.phi * self.delta_L))
                 * (C_NH3_liq - C_NH3_biofilm)
            difusion_O2 = (self.a * self.D_O2 / (self.phi * self.delta_L)) *
                 (C_O2_liq - C_O2_biofilm)
            difusion_NaHCO3 = (self.a * self.D_NaHCO3 / (self.phi * self.
                delta_L)) * (C_NaHCO3 - C_NaHCO3_biofilm)
            difusion_NaNO3 = (self.a * self.D_NaNO3 / (self.phi * self.
                delta_L)) * (C_NaNO3 - C_NaNO3_biofilm)

            # Balances de masa al gas
            dC_NH3_gas_dt = ((self.C_in_NH3_gas - C_NH3_gas) / self.V) -
                self.k_La_NH3 * (C_interfase_NH3 - C_NH3_liq)
            dC_O2_gas_dt = ((self.C_in_O2_gas - C_O2_gas) / self.V) - self.
                k_La_O2 * (C_interfase_O2 - C_O2_liq)

            # Balances de masa al liquido
            dC_NH3_liq_dt = ((self.C_in_NH3_liq - C_NH3_liq) / self.V) +
                self.k_La_NH3 * (C_interfase_NH3 - C_NH3_liq) - difusion_NH3
            dC_NaHCO3_dt = ((self.C_in_NaHCO3 - C_NaHCO3) / self.V) -
                difusion_NaHCO3
            dC_O2_liq_dt = ((self.C_in_O2_liq - C_O2_liq) / self.V) + self.
                k_La_O2 * (C_interfase_O2 - C_O2_liq) - difusion_O2
            dC_NaNO3_dt = ((self.C_in_NaNO3 - C_NaNO3) / self.V) -
                difusion_NaNO3

            # Balances de masa al biofilm
```

```
            dC_NH3_biofilm_dt = difusion_NH3 - eficiencia * mu_biofilm
            dC_NaHCO3_biofilm_dt = difusion_NaHCO3 - eficiencia *
                coef_esteq_NaHCO3 * mu_biofilm
            dC_O2_biofilm_dt = difusion_O2 - eficiencia * mu_biofilm
            dC_NaNO3_biofilm_dt = difusion_NaNO3 + eficiencia *
                coef_esteq_NaNO3 * mu_biofilm
            dC_biofilm_dt = difusion_NH3 + difusion_O2 - eficiencia *
                mu_biofilm

            return [dC_NH3_gas_dt, dC_O2_gas_dt, dC_NH3_liq_dt, dC_NaHCO3_dt
                , dC_O2_liq_dt, dC_NaNO3_dt, dC_NH3_biofilm_dt,
                dC_NaHCO3_biofilm_dt,
                dC_O2_biofilm_dt, dC_NaNO3_biofilm_dt, dC_biofilm_dt]

        # Condiciones iniciales
        'Tomamos una concentracion inicial en el biofilm igual a 0.05. En el
            resto de casos, depende de la etapa'
        C0 = [self.C_in_NH3_gas, self.C_in_O2_gas, self.C_in_NH3_liq, self.
            C_in_NaHCO3, self.C_in_O2_liq, self.C_in_NaNO3, 0.05, 0.05,
            0.05, 0.05, 0.05]

        # Tiempo de simulacion (en horas)
        t = np.linspace(0, 20, 100)  # 20 horas

        # Resolver las ecuaciones diferenciales
        self.solution = odeint(balance_masa, C0, t)
        C_NH3_gas_final, C_O2_gas_final, C_NH3_final, C_NaHCO3_final,
            C_O2_final, C_NaNO3_final, C_NH3_biofilm_final,
            C_NaHCO3_biofilm_final, C_O2_biofilm_final,
            C_NaNO3_biofilm_final, C_biofilm_final = self.solution[-1]

        # Actualizar la corriente de salida
        NH3_out.imass['NH3'] = C_NH3_gas_final * self.V
        NaHCO3_out.imass['NaHCO3'] = C_NaHCO3_final * self.V
        O2_out.imass['O2'] = C_O2_gas_final * self.V
        NaNO3_out.imass['NaNO3'] = C_NaNO3_final * self.V

# Datos

V_total = 2.5 # [L]
N = 54 # Numero de etapas
V_etapa = V_total / N # [L]

'Parametros adicionales del biorreactor'
kwargs = {
    'k_La_NH3': 231612.12,     # 1/h
    'k_H_NH3': 1.0,      # g/L/atm
    'P_NH3': 1.0,        # atm
    'k_La_O2': 231612.12,      # 1/h
    'k_H_O2': 1.2,       # g/L/atm
    'P_O2': 1.0,         # atm
    'v_max': 5.0,        # g/L/h
    'K_m_1': 5.0,       # g/L
    'K_m_2': 5.0,       # g/L
    'delta_L': 0.01,     # [cm]
    'a': 50,            # [cm2]
    'D_NH3': 1e-5,       # [cm2/s]
    'D_O2': 1e-5,        # [cm2/s]
    'D_NaHCO3': 1e-5,    # [cm2/s]
    'D_NaNO3': 1e-5,     # [cm2/s]
    'phi': 0.85
}
```

```
'En␣estas␣listas␣acumularemos␣las␣concentraciones␣para␣posteriormente␣
    representarlas'

C_NH3_gas_etapas = np.zeros(N)
C_O2_gas_etapas = np.zeros(N)
C_NH3_etapas = np.zeros(N)
C_NaHCO3_etapas = np.zeros(N)
C_O2_etapas = np.zeros(N)
C_NaNO3_etapas = np.zeros(N)
C_NH3_biofilm_etapas = np.zeros(N)
C_NaHCO3_biofilm_etapas = np.zeros(N)
C_O2_biofilm_etapas = np.zeros(N)
C_NaNO3_biofilm_etapas = np.zeros(N)
C_biofilm_etapas = np.zeros(N)

'Condiciones␣iniciales'

C_NH3_gas_etapas[0] = 9.120 # g/hr
C_O2_gas_etapas[0] = 16.991 # g/hr
C_NH3_etapas[-1] = 10e-5 # g/hr
C_NaHCO3_etapas[-1] = 4.129 # g/hr por la parte de arriba
C_O2_etapas[-1] = 10e-5 # g/hr
C_NaNO3_etapas[-1] = 0.0 # g/hr
C_NH3_biofilm_etapas[-1] = 9.120
C_NaHCO3_biofilm_etapas[-1] = 4.129 # g/hr
C_O2_biofilm_etapas[-1] = 16.991 # g/hr
C_NaNO3_biofilm_etapas[-1] = 0.05
C_biofilm_etapas[-1] = 0.5

not_convergencia = True # Agregaremos un bucle de convergencia
tol = 1e-6  # Tolerancia de convergencia
max_iter = 100 # M xima iteracion
iter_count = 0 # Recuento iteraciones


while not_convergencia and iter_count < max_iter: # Mientras no converja,
    seguiremos iterando

    iter_count += 1

    # Guardar valores previos para todas las etapas
    prev_C_NH3_gas = C_NH3_gas_etapas.copy()
    prev_C_O2_gas = C_O2_gas_etapas.copy()
    prev_C_NH3 = C_NH3_etapas.copy()
    prev_C_NaHCO3 = C_NaHCO3_etapas.copy()
    prev_C_O2 = C_O2_etapas.copy()
    prev_C_NaNO3 = C_NaNO3_etapas.copy()
    prev_C_NH3_biofilm = C_NH3_biofilm_etapas.copy()
    prev_C_NaHCO3_biofilm = C_NaHCO3_biofilm_etapas.copy()
    prev_C_O2_biofilm = C_O2_biofilm_etapas.copy()
    prev_C_NaNO3_biofilm = C_NaNO3_biofilm_etapas.copy()
    prev_C_biofilm = C_biofilm_etapas.copy()

    zonas = list(range(0, 2)) + list(range(2, 52)) + list(range(52, 53)) #
        Definimos las zonas de las etapas

    for etapa in zonas:

        idx = N - etapa - 1 # Indice para representar las concentraciones en
             sentido contrario
```

```
        biorreactor = biofilter(
            f'Biorreactor_{etapa+1}',
            ins = (NH3_in, NaHCO3_in, O2_in, NaNO3_in),
            outs = (NH3_out, NaHCO3_out, O2_out, NaNO3_out,),
            V = V_etapa,
            C_in_NH3_gas = C_NH3_gas_etapas[etapa],
            C_in_O2_gas = C_O2_gas_etapas[etapa],
            C_in_NH3_liq = C_NH3_etapas[idx],
            C_in_NaHCO3 = C_NaHCO3_etapas[idx],
            C_in_O2_liq = C_O2_etapas[idx],
            C_in_NaNO3 = C_NaNO3_etapas[idx],
            **kwargs
        )

        biorreactor._run()

        solution = biorreactor.solution
        C_NH3_gas_etapas[etapa + 1], C_O2_gas_etapas[etapa + 1],
            C_NH3_etapas[idx - 1], C_NaHCO3_etapas[idx - 1], C_O2_etapas[idx
             - 1], C_NaNO3_etapas[idx - 1], C_NH3_biofilm_etapas[idx - 1],
            C_NaHCO3_biofilm_etapas[idx - 1], C_O2_biofilm_etapas[idx - 1],
            C_NaNO3_biofilm_etapas[idx - 1], C_biofilm_etapas[idx - 1] =
            solution[-1]

        C_NH3_gas_etapas[etapa + 1] = max(C_NH3_gas_etapas[etapa + 1], 0)  #
             Asegurar que las concentraciones no sean negativas
        C_O2_gas_etapas[etapa + 1] = max(C_O2_gas_etapas[etapa + 1], 0)  #
            Asegurar que las concentraciones no sean negativas
        C_NH3_etapas[idx - 1] = max(C_NH3_etapas[idx - 1], 0)  # Asegurar
            que las concentraciones no sean negativas
        C_NaHCO3_etapas[idx - 1] = max(C_NaHCO3_etapas[idx - 1], 0)  #
            Asegurar que las concentraciones no sean negativas
        C_O2_etapas[idx - 1] = max(C_O2_etapas[idx - 1], 0)  # Asegurar que
            las concentraciones no sean negativas
        C_NaNO3_etapas[idx - 1] = max(C_NaNO3_etapas[idx - 1], 0)  #
            Asegurar que las concentraciones no sean negativas
        C_NH3_biofilm_etapas[idx - 1] = max(C_NH3_biofilm_etapas[idx - 1],
            0)
        C_NaHCO3_biofilm_etapas[idx - 1] = max(C_NaHCO3_biofilm_etapas[idx -
             1], 0)
        C_O2_biofilm_etapas[idx - 1] = max(C_O2_biofilm_etapas[idx - 1], 0)
        C_NaNO3_biofilm_etapas[idx - 1] = max(C_NaNO3_biofilm_etapas[idx -
            1], 0)
        C_biofilm_etapas[idx - 1] = max(C_biofilm_etapas[idx - 1], 0)

        NH3_in.imass['NH3'] = C_NH3_gas_etapas[etapa + 1] * V_etapa  #
            Ajustar la masa de NH3
        NaHCO3_in.imass['NaHCO3'] = C_NaHCO3_etapas[idx - 1] * V_etapa  #
            Ajustar la masa de NaHCO3
        O2_in.imass['O2'] = C_O2_gas_etapas[etapa + 1] * V_etapa # Ajustar
            la masa de O2
        NaNO3_in.imass['NaNO3'] = C_NaNO3_etapas[idx - 1] * V_etapa  #
            Ajustar la masa de NaNO3

    # Verificar convergencia global comparando todas las etapas
    for etapa in range(N):
        if (abs(C_NH3_gas_etapas[etapa] - prev_C_NH3_gas[etapa]) > tol or
            abs(C_O2_gas_etapas[etapa] - prev_C_O2_gas[etapa]) > tol or
            abs(C_NH3_etapas[etapa] - prev_C_NH3[etapa]) > tol or
            abs(C_NaHCO3_etapas[etapa] - prev_C_NaHCO3[etapa]) > tol or
            abs(C_O2_etapas[etapa] - prev_C_O2[etapa]) > tol or
            abs(C_NaNO3_etapas[etapa] - prev_C_NaNO3[etapa]) > tol or
```

```
            abs(C_NH3_biofilm_etapas[etapa] - prev_C_NH3_biofilm[etapa]) >
                tol or
            abs(C_NaHCO3_biofilm_etapas[etapa] - prev_C_NaHCO3_biofilm[etapa
                ]) > tol or
            abs(C_O2_biofilm_etapas[etapa] - prev_C_O2_biofilm[etapa]) > tol
                 or
            abs(C_NaNO3_biofilm_etapas[etapa] - prev_C_NaNO3_biofilm[etapa])
                 > tol or
            abs(C_biofilm_etapas[etapa] - prev_C_biofilm[etapa]) > tol):
            converged = False  # Si alguna diferencia es mayor a tol,
                seguimos iterando
            break  # No es necesario seguir revisando m s etapas

D = 0.8 # Diametro del biorreactor [m]
L_total = 1.4
positions= np.linspace(0, L_total, N)

# Crear la figura y los subgraficos
fig, axs = plt.subplots(3, 1, figsize=(8, 12), sharex=True)  # 3 filas, 1
    columna

# Grafica de GAS
axs[0].plot(positions, C_NH3_gas_etapas, label='NH3␣(g/L)', color='b')
axs[0].plot(positions, C_O2_gas_etapas, label='O2␣(g/L)', color='r')
axs[0].set_title('GAS')

# Grafica de LIQUIDO
axs[1].plot(positions, C_NH3_etapas, label='NH3␣(g/L)', color='b')
axs[1].plot(positions, C_NaHCO3_etapas, label='NaHCO3␣(g/L)', color='g')
axs[1].plot(positions, C_O2_etapas, label='O2␣(g/L)', color='r')
axs[1].plot(positions, C_NaNO3_etapas, label='NaNO3␣(g/L)', color='m')
axs[1].set_title('L QUIDO')

# Grafica de BIOFILM (todas las concentraciones)
axs[2].plot(positions, C_NH3_biofilm_etapas, label='NH3␣Liquido␣(g/L)',
    color='b')
axs[2].plot(positions, C_NaHCO3_biofilm_etapas, label='NaHCO3␣(g/L)', color=
    'g')
axs[2].plot(positions, C_O2_biofilm_etapas, label='O2␣Liquido␣(g/L)', color=
    'r')
axs[2].plot(positions, C_NaNO3_biofilm_etapas, label='NaNO3␣(g/L)', color='m
    ')
axs[2].plot(positions, C_biofilm_etapas, label='Biofilm␣(g/L)', color='
    orange')
axs[2].set_title('BIOFILM')

# Etiquetas y leyendas
for ax in axs:
    ax.set_ylabel('Conc.␣(g/L)')
    ax.legend()
    ax.grid(True)

axs[-1].set_xlabel('Posicion␣a␣lo␣largo␣del␣PFR␣(m)')  # Solo la ultima
    grafica

# Ajustar los espacios entre las graficas
plt.tight_layout()

# Mostrar la figura
plt.show()
```

Listing B.1: Variación de concentraciones axiales

Anexo C

Modelo fluidodinámico 2D

Con el objetivo de determinar el gradiente de presión necesario para generar un caudal másico específico a través del medio poroso del biorreactor, se ha realizado previamente una simulación en dos dimensiones (2D) mediante ANSYS Fluent. Esta aproximación simplificada permite obtener de forma rápida y eficiente una primera estimación del comportamiento hidráulico del sistema, reduciendo el tiempo computacional respecto a un modelo completo en tres dimensiones (3D).

C.1. Geometría

Para la geometría 2D del volumen se ha representado una posible sección transversal del cubo, dando lugar a un cuadrado de dimensiones **29 mm x 26 mm** con un hueco circular en el centro de **10.2 mm** de diámetro, simulando la sección del anillo Kaldnes.

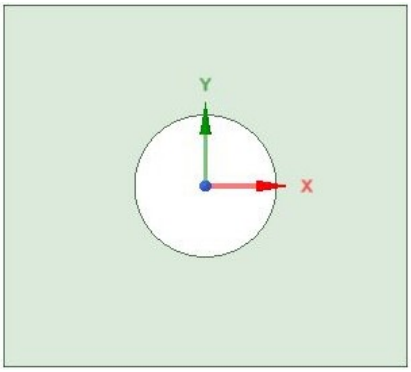

Figura C.1: Geometría 2D

C.2. Mallado

El mallado se realizará con cuadrilateros, los cuales, en comparación con las celdas triangulares, tienen mejor alineación de las caras respecto al flujo principal, lo que mejora la precisión de los gradientes calculados en cada celda. Además, necesitan menos elementos para mallar con una calidad numérica aceptable, reduciendo así el tiempo de cálculo y el consumo de memoria. Las celdas triangulares se suelen utilizar en geometrías complejas e irregulares, y este no es nuestro caso.

El tamaño de malla fue especificado imponiendo 200 divisiones alrededor de la circunferencia interior. Además, se realizó un refinamiento de la zona alrededor del hueco, ya que se considera que el líquido se depositará alrededor del anillo, creando una pequeña película debido a la poca cantidad que entra en el biorreactor. Como las dimensiones de dicha película serán ínfimas, necesitamos un ajuste muy fino para poder observar cambios. La malla se puede observar en la figura C.2.

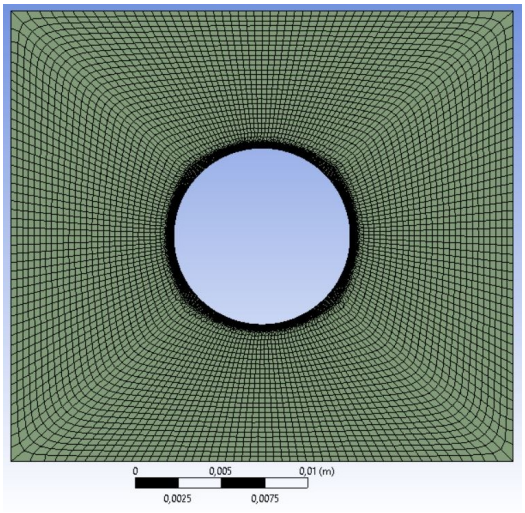

Figura C.2: Mallado 2D

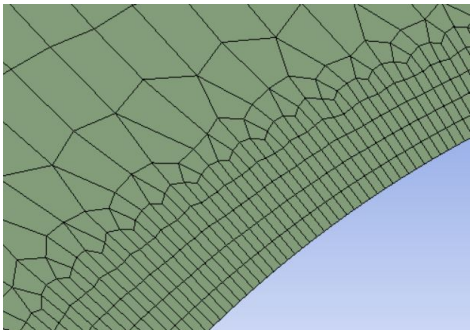

Figura C.3: Detalle del refinamiento de malla

C.3. Condiciones de contorno

En los contornos laterales del dominio bidimensional se ha aplicado una condición de contorno de tipo *symmetry*. Esta elección se fundamenta en que, al tratarse de una representación bidimensional de un volumen representativo

de medio poroso, se puede asumir que las condiciones de contorno en dichas caras son estadísticamente idénticas a las de las zonas adyacentes.

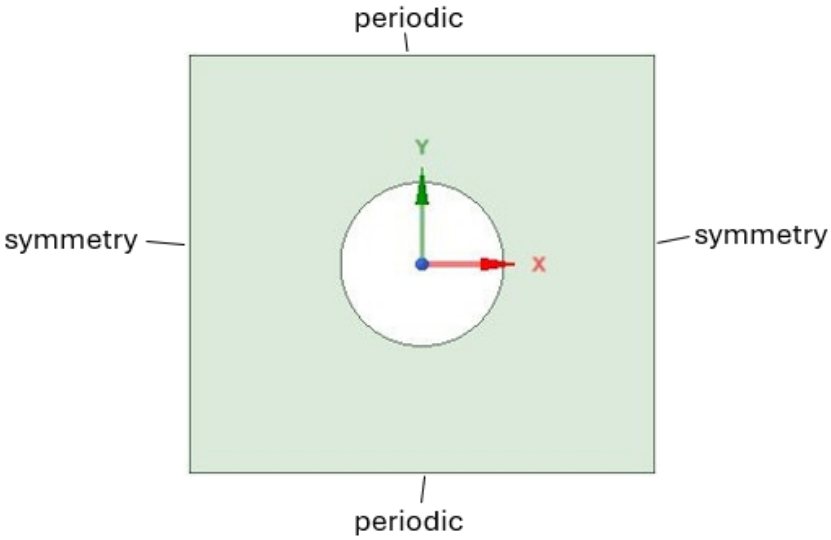

Figura C.4: Condiciones de contorno en 2D

La condición *symmetry* impone que no exista flujo normal ni transferencia de cantidad de movimiento a través del contorno, permitiendo de este modo simular de forma simplificada una sección infinita o periódica del medio. Además, esta simplificación contribuye a reducir el coste computacional sin afectar significativamente a la precisión de los resultados en el interior del dominio, donde se centra el análisis de caída de presión.

Por otra parte, se impone una condición periódica entre la entrada (superficie inferior) y la salida (superficie superior), la cual se utiliza para simular que el flujo que sale vuelve a entrar por la cara inlet con una diferencia de presión impuesta. Esto permite modelar una sección representativa de un medio repetitivo como si fuera parte de un dominio infinito o cíclico, reduciendo notablemente el tamaño del dominio necesario para obtener resultados realistas.

C.4. Cálculo de la caída de presión

La diferencia de presión que se impone se obtiene de manera iterativa, hasta que alcanzamos el valor del caudal que nos ha sido proporcionado por los

colaboradores.

Si bien es cierto que el flujo másico de gas para todo el biorreactor es de 0.168 kg/s (o 0.14 m^3/s si nos referimos a caudal volumétrico), debemos adaptar dicho caudal al área de paso con el que nos encontramos. El modelo real se trata de un biorreactor cilíndrico con un área de paso de 0.5 m^3. Modelamos un volumen representativo tridimensional como un cubo cuya anchura y profundidad son ambas de 0.029 m, mientras que la altura es de 0.026 m, mientras que en el modelo bidimensional se considera una profundidad por defecto de 1 m. La razón de proporcionalidad, en función de dicho área de paso, se describe a continuación:

Modelo	Área $[m^2]$	Razón de proporcionalidad
Real	0.5	-
2D	0.029	17.24
3D	0.000841	594.53

Tabla C.1: Relación entre modelos

Teniendo estas medidas en cuenta, el gradiente de presión adecuado toma un valor de **847 Pa/m**. Los flujos másicos obtenidos, en kg/s, tanto teóricos como en simulación, y tanto para el gas como para el líquido, se recogen en la tabla C.2.

Modelo	$\dot{M}_{G,T}$	$\dot{M}_{G,S}$	$\dot{M}_{L,T}$	$\dot{M}_{L,S}$
Real	0.168	-	9.89	-
2D	0.00974	0.00994	0.5736	0.5625
3D	0.000282	0.000315	0.01662	0.01753

Tabla C.2: Flujos másicos

También es interesante comparar los flujos volumétricos, en m^3/s, tal y como se puede observar en la tabla C.3.

C.5. Resultados

A partir de las simulaciones realizadas en el modelo bidimensional, se han obtenido los mapas de distribución de velocidad y presión en régimen permanente para las condiciones de contorno definidas. Estos resultados permiten comprobar el comportamiento del flujo al atravesar la zona ocupada por el obstáculo (representando los anillos Kaldnes) y validar que la diferencia de

Modelo	$Q_{G,T}$	$Q_{G,S}$	$Q_{L,T}$	$Q_{L,S}$
Real	0.14	-	0.00991	-
2D	0.00812	0.008116	0.0005748	0.0005635
3D	0.000235	0.000257	$1{,}666 \cdot 10^{-5}$	$1{,}757 \cdot 10{-}5$

Tabla C.3: Flujos volumétricos

presión alcanzada es coherente con los caudales másicos teóricos previamente calculados.

C.5.1. Campo de velocidad

Se representa el módulo de velocidad (velocity magnitude), que es la magnitud escalar de la velocidad en cada punto, se mide en m/s.

$$|\vec{v}| = \sqrt{u^2 + v^2} \tag{C.1}$$

La figura correspondiente a los contornos de esta velocidad es la siguiente:

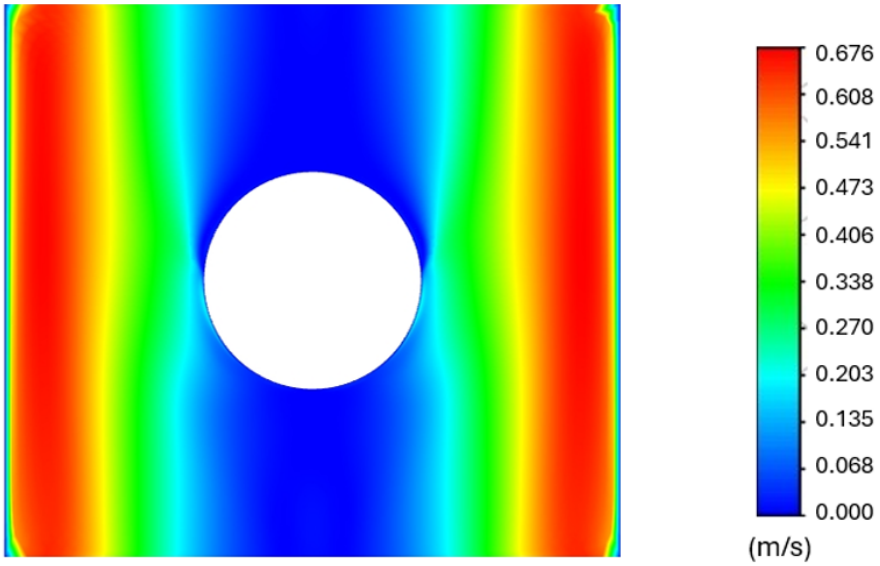

Figura C.5: Contornos del módulo de velocidad

En esta figura se observa el efecto de la presencia del hueco central en el comportamiento del flujo dentro del dominio. A la entrada del dominio, el perfil de velocidades es prácticamente uniforme, debido a las condiciones de contorno impuestas, con una magnitud máxima cercana a **0.676 m/s**.

Al aproximarse al hueco central, el flujo se ve obligado a rodearlo, generando una aceleración en las zonas cercanas a los bordes laterales del hueco, mientras que en su parte frontal y posterior la velocidad disminuye notablemente, llegando a valores próximos a cero e incluso negativos, debido al cambio de dirección que debe experimentar el flujo.

Se aprecia también cómo en los contornos laterales del dominio la velocidad normal es nula, consecuencia directa de la condición de contorno *Symmetry* aplicada en dichos bordes. Esta condición impide la existencia de flujo atravesando los planos laterales.

Finalmente, tras bordear el hueco, el flujo recupera progresivamente su magnitud inicial hacia la zona de salida, aunque con una ligera asimetría provocada por las perturbaciones generadas alrededor del obstáculo.

C.5.2. Campo de presiones

Representamos, en concreto, la presión estática por ser la que muestra las pérdidas de carga, el gradiente a través del medio poroso y las condiciones reales de presión que afectan al flujo y a la transferencia de masa.

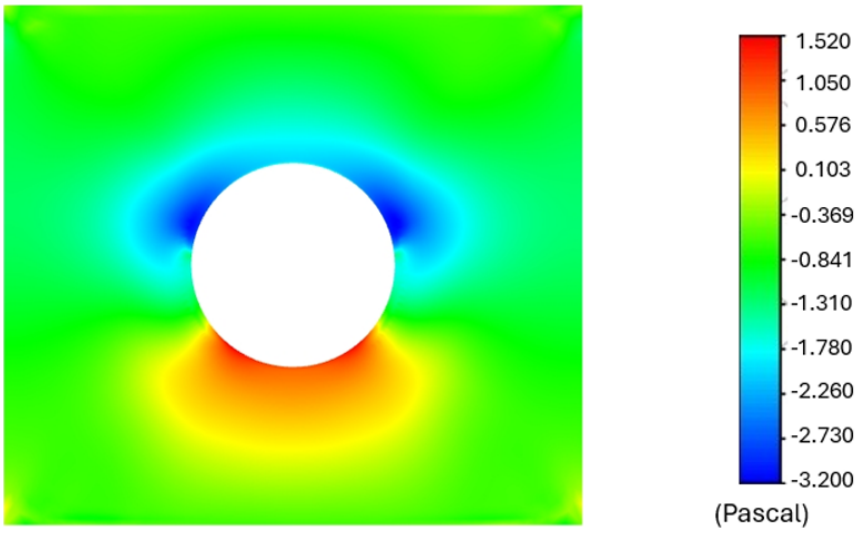

Figura C.6: Contornos de presión estática

En la figura C.6 se aprecia cómo la distribución de presión en el interior del dominio está condicionada por la presencia del hueco central y por las con-

diciones de contorno impuestas.

A la entrada del dominio, la presión presenta su valor máximo, correspondiente a la condición de entrada, y se mantiene prácticamente uniforme en esa región. Conforme el flujo se aproxima al hueco central, se produce un incremento de presión en la zona frontal del obstáculo, debido a la disminución de área de paso y al efecto de frenado que sufre el fluido. Esta sobrepresión es característica de la interacción entre un flujo y una obstrucción fija.

Por el contrario, en la región situada justo detrás del hueco central se forma una zona de baja presión. Este comportamiento se debe a la expansión súbita del flujo al superar el obstáculo, generando una estela de presión reducida aguas abajo.

El campo de presión resultante es coherente con el comportamiento esperado en un flujo alrededor de una obstrucción, y permite validar que el modelo reproduce correctamente los fenómenos de pérdida de carga y distribución de presiones que posteriormente se trasladarán al modelo tridimensional.

C.5.3. Líneas de corriente

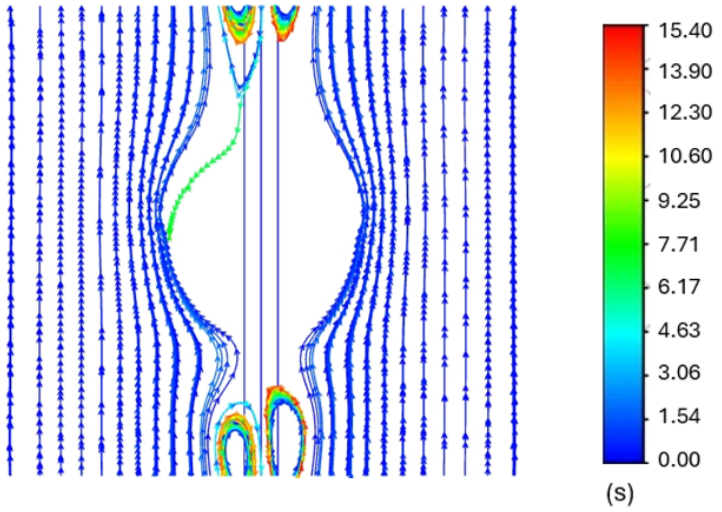

Figura C.7: Líneas de corriente en el modelo 2D

También es interesante analizar las líneas de corriente, pues simulan cómo se mueve el fluido en el dominio en un instante dado. Es una potente herramienta para visualizar zonas de turbulencia, zonas muertas, zonas donde el

fluido se acelera o donde se estanca, etc.

En la imagen C.7 podemos observar, por un lado, que el fluido se frena al pasar por las inmediaciones del hueco central, que simula el obstáculo, ya que la condición en la pared es de velocidad nula.

Por último, vemos que el sentido prioritario del flujo es ascendente, como es lógico ya que el medio continuo es de gas, que se inyecta en la parte inferior y asciende venciendo a la fuerza gravitatoria debido a un gradiente de presión impuesto.

Anexo D

Análisis de la Configuración del Líquido: Formación de Gotas o Película en el Medio Poroso

Tal y como se comenta en el capítulo 4, no podemos asumir directamente el comportamiento del líquido en las paredes de los anillos. Es por esto que se ha llevado a cabo un estudio del estado en el que se encuentra el agua, comparando el espesor de la película con el diámetro de las gotas, pues la forma en que el líquido se distribuye sobre el medio poroso influye directamente en el área interfacial disponible para la transferencia de oxígeno y en el valor de $k_L a$.

Calculamos el espesor de la película, δ, de líquido que se formaría sobre la pared mojada. Para ello, utilizamos una correlación de tipo Nusselt para película laminar descendente por gravedad [40].

$$\delta = (\frac{3 \cdot \mu_L \cdot Q_L}{\rho_L \cdot g \cdot P_{mojado} \cdot sin(\theta)})^{1/3} \tag{D.1}$$

Donde:

- μ_L es la viscosidad dinámica del agua: 0.001 $kg/(m \cdot s)$
- Q_L es el caudal de líquido: $1{,}666 \cdot 10^{-5}$ m^2/s
- ρ_L es la densidad del líquido: 998 kg/m^3
- g es la gravedad: 9,81 m/s^2
- P_{mojado} es el perímetro mojado, que consideramos que es la altura del anillo: 0,01 m

- θ es el ángulo de inclinación: 11.25°

Sabemos que el caudal de líquido que atraviesa el biorreactor es de $9{,}91 \cdot 10^{-3}\ m^3/s$. Acudiendo a la tabla C.1, sabemos que la razón de proporcionalidad entre el modelo real y el volumen representativo 3D es de 594.53. Con esto, ya podemos calcular el caudal de líquido, Q_L.

$$Q_L = \frac{9{,}91 \cdot 10^{-3}}{594{,}53} = 1{,}666 \cdot 10^{-5}\ m^3/s \tag{D.2}$$

Consideramos que el perímetro mojado es la altura del anillo, ya que es el recorrido vertical del líquido, tal y como observamos en la imagen D.1.

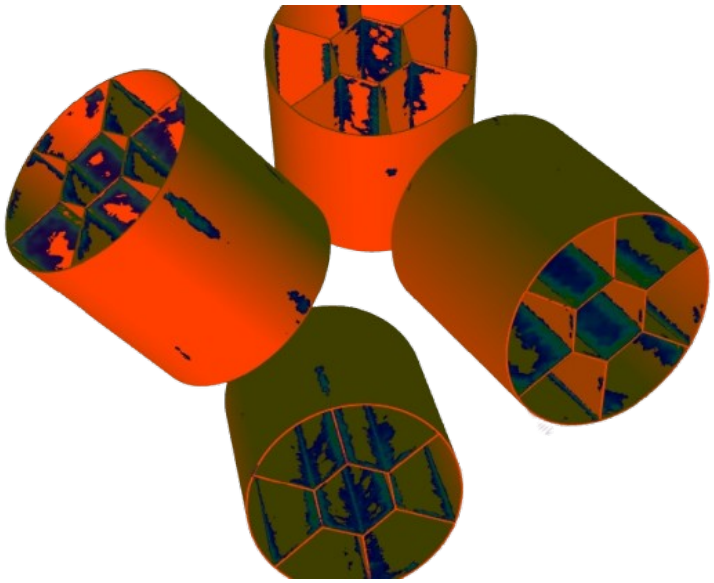

Figura D.1: Área mojada

Por último, el ángulo de inclinación con la vertical en dirección de la gravedad es distinto para cada anillo, por lo que utilizamos un promedio:

$$\bar{\theta} = \frac{15° + 15° + 0° + 15°}{4} = 11{,}25° \tag{D.3}$$

Con todo esto, ya podemos calcular el espesor de la película, δ.

$$\delta = \left(\frac{3 \cdot 0{,}001[kg/(m \cdot s)] \cdot 1{,}666 \cdot 10^{-5}[m^3/s]}{998[kg/m^3] \cdot 9{,}81[m/s^2] \cdot 0{,}01[m] \cdot sin(11{,}25°)}\right)^{1/3} = 1{,}378 \cdot 10^{-3}\ m \tag{D.4}$$

Es decir, la película tiene un espesor de 1.378 mm > diámetro de gota (alrededor de 0.1mm). Por lo tanto, la película es lo suficientemente ancha para considerar que se forma en las paredes del anillo.

Anexo E

Código en Python para DTR y gráfica de $k_L \cdot a \cdot \tau$ vs Re

E.1. Gráfica de DTR en el modelo cinético

Se implementa el código realizado en Python para obtener la gráfica de Distribución de Tiempos de Residencia dentro del modelo cinético

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

' PAR METROS DEL SISTEMA '
n = 53                                  # n   de CSTRs en serie
Q = 140                                 # Caudal volum trico [L/s]
V_total = 700                           # Volumen total del PFR [L]
V = V_total / n                         # Volumen de cada reactor [L]
tau = V / Q                             # Tiempo de residencia medio de cada
    reactor [s]
t_max = 5 * n * tau                     # Tiempo m ximo de simulaci n
t_eval = np.linspace(0, t_max, 500) # Puntos de tiempo para evaluaci n

' FUNCI N BALANCE DE MASA EN LOS CSTRs '
def CSTR(t, C):
    dCdt = np.zeros(n)
    dCdt[0] = -C[0] / tau
    for i in range(1, n):
        dCdt[i] = (C[i-1] - C[i]) / tau
    return dCdt

' CONDICIONES INICIALES (pulso del trazador) '
C0 = np.zeros(n)
C0[0] = 1.0         # Pulso en el primer reactor

' RESOLVER Y GRAFICAR '
sol = solve_ivp(CSTR, [0, t_max], C0, t_eval = t_eval)

plt.figure(figsize=(8, 5))
plt.plot(sol.t, sol.y[-1], label=f"n = {n} CSTRs", linewidth=2)
```

```
plt.xlabel("Tiempo␣[s]")
plt.ylabel("Concentraci n␣del␣trazador")
plt.title("Distribuci n␣de␣Tiempos␣de␣Residencia␣(RTD)")
plt.legend()
plt.grid()
plt.show()
```

Listing E.1: DTR con 53 CSTR

E.2. Gráfica de DTR en el modelo fluidodinámico

Para calcular los tiempos de residencia en estado estacionario a través de Fluent se utilizaron las llamadas *pathlines* o líneas de corriente: trayectorias de partículas de la mezcla que muestran el tiempo acumulado hasta su salida, tal y como se observa en la figura E.1.

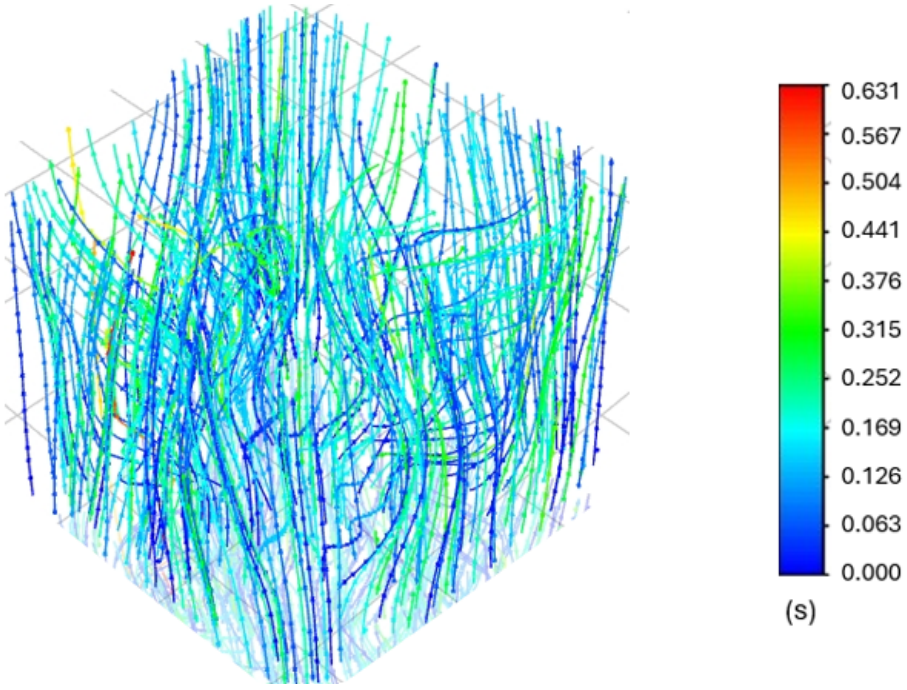

Figura E.1: Líneas de corriente

Se obtuvieron los valores de los tiempos de salida de cada partícula, que coincidían con el último tiempo registrado, ya que son acumulativos. Con esto, se calculó el tiempo medio y la curva E(t).

E.2.1. Código en Python

```
import numpy as np
import matplotlib.pyplot as plt

# Lista de tiempos de residencia
tiempos = np.array([-]) #Rellenar con tiempos para cada caso

# Histograma base
cuentas, bordes = np.histogram(tiempos, bins=20, density=False)
delta_t = bordes[1] - bordes[0]
E_t = cuentas / (len(tiempos) * delta_t)
t = 0.5 * (bordes[:-1] + bordes[1:])

# Tiempo medio base
t_medio = np.sum(t * E_t) * delta_t
print(f"Tiempo␣medio␣de␣residencia␣base:␣{t_medio:.4f}␣s")

# Bootstrap para estimar incertidumbre
N_iter = 1000
t_medios = []

for _ in range(N_iter):
    muestra = np.random.choice(tiempos, size=len(tiempos), replace=True)
    cuentas_m, _ = np.histogram(muestra, bins=bordes, density=False)
    E_t_m = cuentas_m / (len(muestra) * delta_t)
    t_m = np.sum(t * E_t_m) * delta_t
    t_medios.append(t_m)

t_medios = np.array(t_medios)

# Estad sticos
media_bootstrap = np.mean(t_medios)
desv_std_bootstrap = np.std(t_medios, ddof=1)

# IC 95% aproximado asumiendo normalidad (media    2  )
IC_95_aprox = [media_bootstrap - 2 * desv_std_bootstrap, media_bootstrap + 2
    * desv_std_bootstrap]

# Resultados en consola
print(f"\nTiempo␣medio␣estimado␣(bootstrap):␣{media_bootstrap:.4f}␣s")
print(f"Desviaci n␣est ndar␣(bootstrap):␣{desv_std_bootstrap:.4f}␣s")
print(f"IC␣95%␣aprox␣(media␣  ␣2  ):␣[{IC_95_aprox[0]:.4f},␣{IC_95_aprox
    [1]:.4f}]␣s")

# Graficar RTD original
plt.plot(t, E_t, label='RTD␣E(t)')
plt.xlabel('Tiempo␣[s]')
plt.ylabel('E(t)␣[1/s]')
plt.title('Distribuci n␣de␣Tiempos␣de␣Residencia')
plt.grid(True)
plt.axvline(t_medio, color='r', linestyle='--', label=f'Tiempo␣medio␣base␣=␣
    {t_medio:.3f}␣s')
plt.axvspan(IC_95_aprox[0], IC_95_aprox[1], color='yellow', alpha=0.2, label
    ='IC␣95%␣Aprox␣  2  ')
plt.legend()
plt.show()

# Histograma de tiempos medios bootstrap
plt.hist(t_medios, bins=30, color='skyblue', edgecolor='k')
```

```
plt.axvline(media_bootstrap, color='r', linestyle='--', label=f'Media␣
    Bootstrap␣=␣{media_bootstrap:.3f}␣s')
plt.axvline(IC_95_aprox[0], color='orange', linestyle='--', label=f'IC␣95%␣
    Aprox␣  2  ')
plt.axvline(IC_95_aprox[1], color='orange', linestyle='--')
plt.xlabel('Tiempo␣medio␣de␣residencia␣[s]')
plt.ylabel('Frecuencia')
plt.title('Distribuci n␣Bootstrap␣del␣Tiempo␣Medio')
plt.legend()
plt.grid(True)
plt.show()
```

Listing E.2: Cálculo de E(t)

E.3. Gráfica de $k_L \cdot a$. Código en Python

E.3.1. Código para distribuciones aleatorias

```
    import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Datos de entrada
x = [31, 47, 62, 93, 104, 117, 133, 156, 187, 233, 311, 467]
y = [2.0647, 0.8045, 0.4478, 0.2875, 0.2381, 0.1946, 0.1641, 0.1385, 0.1179,
     0.0712, 0.0385, 0.0119]
# Par metros
n_samples = 1000
std_dev = 0.061

# Inicializar listas
means = []
lower_bounds = []
upper_bounds = []

# Calcular estad sticas
for yi in y:
    samples = np.random.normal(loc=yi, scale=std_dev, size=n_samples)
    mean = np.mean(samples)
    lower = np.percentile(samples, 2.5)
    upper = np.percentile(samples, 97.5)
    means.append(mean)
    lower_bounds.append(lower)
    upper_bounds.append(upper)

# Convertir a arrays
means = np.array(means)
lower_bounds = np.array(lower_bounds)
upper_bounds = np.array(upper_bounds)

# Mostrar tabla con pandas
df = pd.DataFrame({
    'x': x,
    'Media': means,
    'IC␣2.5%␣(Inferior)': lower_bounds,
    'IC␣97.5%␣(Superior)': upper_bounds
})
```

```
print("\nTabla de valores con intervalo de confianza del 95%:")
print(df.to_string(index=False))

# Graficar
plt.plot(x, means, '-o', color='blue', label='Media')
plt.fill_between(x, lower_bounds, upper_bounds, color='blue', alpha=0.2,
                 label='95% intervalo de confianza')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Distribuciones aleatorias con incertidumbre (95%)')
plt.grid(True)
plt.legend()
plt.show()
```

Listing E.3: Distribuciones aleatorias de $k_L \cdot a$

E.3.2. Código para la gráfica

```
import matplotlib.pyplot as plt
import numpy as np

# Datos (invertidos)
x = np.array([31, 47, 62, 93, 104, 117, 133, 156, 187, 233, 311, 467])
y = np.array([2.065, 0.804, 0.448, 0.287, 0.238, 0.195, 0.164, 0.138, 0.118,
     0.071, 0.039, 0.012])
y_min = np.array([1.945, 0.695, 0.334, 0.163, 0.109, 0.077, 0.036, 0.019,
    0.000, 0.000, 0.000, 0.000])
y_max = np.array([2.188, 0.926, 0.564, 0.400, 0.352, 0.310, 0.276, 0.257,
    0.238, 0.192, 0.160, 0.130])

# Opciones de divisiones
x_ticks = 10  # Cambia este n mero seg n lo necesites
y_ticks = 10  # Igual aqu

# Crear gr fico
fig, ax = plt.subplots()

# Sombrear el rango de incertidumbre
ax.fill_between(x, y_min, y_max, color='blue', alpha=0.3, label='Rango de
    incertidumbre')

# L nea principal
ax.plot(x, y, color='blue', marker='o', label='kL a medio')

# Marcas azules para extremos de incertidumbre
ax.plot(x, y_min, linestyle='None', marker='_', color='blue')
ax.plot(x, y_max, linestyle='None', marker='_', color='blue')

# Control de divisiones en los ejes
ax.locator_params(axis='x', nbins=x_ticks)
ax.locator_params(axis='y', nbins=y_ticks)

# Etiquetas
ax.set_xlabel('N mero de Reynolds')
ax.set_ylabel('kL a [1/s]')
ax.set_title('kL a vs Re')
ax.legend()
ax.grid(True)
```

```
plt.tight_layout()
plt.show()
```

Listing E.4: Gráfica de k_La

E.4. Gráfica de $k_La \cdot \tau$ vs Re

Multiplicamos los valores de k_La medio y τ medio, así como de sus rangos de incertidumbre.

```
import numpy as np
import matplotlib.pyplot as plt

# Datos
Re = np.array([467, 311, 233, 187, 156, 133, 117, 104, 93, 62, 47, 31])
kL = np.array([0.00797, 0.00749, 0.00719, 0.00707, 0.00668, 0.00656,
               0.00640, 0.00628, 0.00623, 0.00616, 0.00611, 0.00602])
a = np.array([1.50, 5.15, 9.90, 16.68, 20.73, 25.03,
              30.40, 37.93, 46.15, 72.66, 131.63, 342.84])
tau = np.array([72.01, 53.51, 45.67, 34.75, 36.97, 35.25,
                23.21, 19.42, 14.29, 13.95, 10.13, 6.91])
tau_lower = np.array([67.3, 50.60, 41.45, 30.69, 27.45, 41.45,
                      18.84, 18.30, 13.46, 11.84, 9.15, 6.46])
tau_upper = np.array([76.45, 56.53, 49.53, 39.30, 46.84, 50.60,
                      27.46, 20.46, 15.07, 16.15, 10.77, 7.54])

# Producto
product = kL * a * tau

# Incertidumbre relativa de kL
kL_err = 0.00002
kL_rel_err = kL_err / kL

# Incertidumbre relativa de tau (usar la peor de cada lado)
tau_err_lower = tau - tau_lower
tau_err_upper = tau_upper - tau
tau_rel_err = np.maximum(tau_err_lower, tau_err_upper) / tau

# Incertidumbre total relativa (a no tiene incertidumbre)
total_rel_err = np.sqrt(kL_rel_err**2 + tau_rel_err**2)

# Incertidumbre absoluta
product_err = product * total_rel_err

# L mites
product_lower = np.maximum(product - product_err, 0)
product_upper = product + product_err

# Graficar
plt.figure(figsize=(10, 6))
plt.fill_between(Re, product_lower, product_upper, color='lightcoral', alpha
    =0.4, label='Incertidumbre')
plt.plot(Re, product, 'o-', color='red', label='kL␣ ␣a␣ ␣tau')
plt.plot(Re, product_lower, '_', color='red', markersize=12, label='L mites
    ␣inferiores')
plt.plot(Re, product_upper, '_', color='red', markersize=12, label='L mites
    ␣superiores')
```

```
plt.xlabel('N mero de Reynolds')
plt.ylabel('kL  a  tau')
plt.title('Producto kL  a  tau con incertidumbre')
plt.grid(True, linestyle='--', alpha=0.6)
plt.xticks(np.arange(0, max(Re)+50, 50))
plt.yticks(np.linspace(0, np.max(product_upper)*1.1, 20))
plt.legend()
plt.tight_layout()
plt.show()
```

Listing E.5: Gráfica de $k_L a \cdot \tau$ vs Re