



FUNDACIÓN UNIVERSITARIA KONRAD LORENZ

**EFICACIA DE LAS ECUACIONES DIFERENCIALES NEURONALES EN UN MODELO DE CRECIMIENTO
TUMORAL CON RETARDO**

Nathalia Valentina Castiblanco Carretero

Trabajo de grado para optar por el título de:

Matemática

John Alexander Arredondo Garcia

Director de trabajo de grado

Miguel González Duque

Codirector de trabajo de grado

Programa de Matemáticas

Facultad de Matemáticas e Ingeniería

Bogotá D.C. Mayo de 2024

Dedicatoria

A mi familia.

En especial a mis padres por su inquebrantable apoyo, su constante aliento y su amor incondicional a lo largo de este viaje académico. Gracias por ser mis pilares, por creer en mí incluso cuando yo dudaba, y por ser mi fuente inagotable de inspiración. Este logro es también suyo.

A mi pareja

Por ser mi compañero de vida y una gran motivación. Tus palabras de aliento, tu comprensión y tu amor han sido fundamentales en este camino hacia la culminación de mi trabajo de grado. Gracias por estar siempre a mi lado, celebrando mis triunfos y apoyándome en los desafíos.

Agradecimientos

Quiero expresar mi más sincero agradecimiento a la Fundación Universitaria Konrad Lorenz por brindarme la oportunidad de formarme académicamente y por su constante apoyo durante mi proceso de estudios.

Agradezco especialmente a mis profesores por su dedicación y orientación a lo largo de mi carrera. En particular, quiero reconocer el invaluable apoyo del profesor Alexander Arredondo, quien dirigió mi trabajo de grado y me proporcionó orientación experta, motivación y aliento durante todo el proceso de investigación. Su compromiso y conocimiento fueron fundamentales para el desarrollo y éxito de este trabajo.

Asimismo, quiero extender mi gratitud a Miguel Duque, quien, a pesar de no estar vinculado directamente con la universidad, aceptó ser mi codirector en este trabajo. Su visión, asesoramiento y disposición para colaborar fueron de gran ayuda y contribuyeron significativamente a la realización de este proyecto.

Agradezco también a todas las personas que de una forma u otra contribuyeron a este trabajo, así como a mis familiares, pareja y amigos por su apoyo incondicional a lo largo de esta etapa académica.

Índice general

Dedicatoria	II
Agradecimientos	III
Resumen	XI
1. Introducción	1
2. Un modelo de crecimiento tumoral en ecuaciones diferenciales con y sin retraso	4
2.1. Antecedentes biológicos y origen del modelo.	4
2.2. Análisis dinámico del modelo sin retardo	8
2.2.1. Estabilidad para el estado estacionario P_0	11
2.2.2. Estabilidad para el estado estacionario P^*	12
2.2.3. Estabilidad para el estado estacionario P_*	15
2.2.4. Estabilidad para el estado estacionario P_*^*	17
2.2.5. Estabilidad global de los estados estacionarios	18
2.3. Análisis de estabilidad del modelo con retardo temporal.	23
2.3.1. Estabilidad local del modelo (2.6).	23
2.3.2. Estabilidad global del modelo (2.6)	28
3. Simulaciones numéricas	38
4. Introducción a las ecuaciones diferenciales neurales	47
4.1. Redes neuronales de profundidad continua:	49
4.2. Entrenamiento de las NDEs	50
4.2.1. Retropropagando a través del solucionador de ODEs	51

4.2.2. Método de la adjunta	52
4.2.3. Solucionador de ODE: Tsit5() y Controlador PID	53
4.2.4. Optimizador: optax.adabelief	53
4.3. Potenciales aplicaciones	54
4.3.1. Ventajas	54
4.3.2. Desventajas	55
5. Implementación del Modelo de Crecimiento Tumoral con NeuralODEs	56
5.1. Implementación con datos sin retardo	57
5.1.1. Generación de Datos	57
5.1.2. Arquitectura de la Red	58
5.1.3. Entrenamiento de la NeuralODE	61
5.1.4. Resultados de la NeuralODE	63
5.1.5. Implementación de la RNN	64
5.1.6. Análisis de Resultados y de Errores	67
5.2. Implementación con datos con retardo	69
5.2.1. Generación de Datos	70
5.2.2. Análisis de Resultados y Comparación de Errores	71
5.3. Introducción a las NDDEs	73
5.3.1. Método de Sensibilidad Adjunto para las NDDEs	73
5.3.2. Algoritmo de Solución	74
6. Conclusiones	75
A. Análisis de Estacionalidad en Sistemas Dinámicos	78
A.1. Punto de Equilibrio	78
A.2. Estabilidad Local	78
A.2.1. Análisis de la Matriz Jacobiana	79
A.2.2. Análisis de Estabilidad	79
A.2.3. Solución de la Ecuación Cuadrática	80
A.3. Estabilidad Global y Funciones de Lyapunov	81
A.3.1. Funciones de Lyapunov	81
A.4. Bifurcación de Hopf	82

A.4.1. Condiciones para una Bifurcación de Hopf	82
B. Introducción a las Redes Neuronales Recurrentes y la Celda GRU	83
B.1. Redes Neuronales Recurrentes (RNN)	83
B.1.1. La Celda GRU	84
C. Implementaciones	86
Referencias	89

Índice de figuras

3.1. La órbita periódica estable del modelo (2.6) a medida que $\mu K > \eta$ y $n\sigma < r\eta$. Aquí, $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 4$, $\eta = 1,5$ y $\tau = 0,3 > 0,2242 = \tau_0$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0,3272, 2,6138)$, y una órbita periódica estable.	39
3.2. Las órbitas del modelo (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 0,25, 0,2, 0,1, 0$ en el caso 1 de (2.1). Aquí, $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 4$, $\eta = 1,5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0,3272, 2,6138)$, y una órbita periódica estable.	40
3.3. Diagrama de bifurcación del modelo (2.6) Diagrama de bifurcación del modelo (2.6) con respecto al retardo τ cuando $\mu K > \eta$ y $n\sigma < r\eta$. Aquí, excepto para el retardo τ , los valores de otros parámetros son los mismos que los de la Figura (3.1).	41
3.4. Las trayectorias de $T = T(t)$ y $E = E(t)$ del modelo (2.6) correspondientes a tres retardos $\tau = 1,5, 10$ en caso de $0 < \mu < \frac{\eta}{K}$ y $n\sigma < r\eta$, respectivamente. Otros valores de los parámetros son $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 0,4$ y $\eta = 1,5$.	41
3.5. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10,5, 2, 0$ en el caso $\mu < 0$, $\eta + \mu K \geq 0$, $n\sigma < r\eta$. Aquí, $r = 2,5$, $K = 0,5$, $n = 0,8$, $\sigma = 0,5$, $\mu = -0,5$, $\eta = 1,5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0,454, 0,290)$, y una órbita periódica estable.	42
3.6. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10,5, 2, 0$ en el caso $\mu < 0$, $\eta + \mu K < 0$, $n\sigma \leq r\eta$. Aquí, $r = 2,5$, $K = 3,0$, $n = 0,8$, $\sigma = 0,5$, $\mu = -2,5$, $\eta = 1,5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(2,946, 0,056)$, y una órbita periódica estable.	43

3.7.	Las trayectorias del modelo (2.6) con dos retardos $\tau = 0.3, 2.0$ en el Caso C4 de la Tabla (2.1), donde la columna de la izquierda es para $\tau = 0,3$, la columna de la derecha es para $\tau = 2,0$, y las condiciones iniciales del modelo (2.6) son las mismas en las dos columnas. La primera fila son las órbitas del modelo (2.6) en el plano $T - E$, la segunda las trayectorias de $T = T(t)$, y la tercera las de $E = E(t)$	44
3.8.	El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a tres retardos de tiempo $\tau = 10, 5, 0$ en el caso $\mu > 0, n\sigma \geq r\eta$. Aquí, $r = 0,5, K = 2,0, n = 1,8, \sigma = 2,5, \mu = 4,0, \eta = 1,5$. El modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 1,667)$	45
3.9.	El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$ en el caso $\mu < 0, \eta + \mu K < 0, n\sigma > -\frac{rK}{4\mu}(\mu - \frac{\eta}{K})^2$. Aquí, $r = 1,5, K = 2,0, n = 0,9, \sigma = 0,5, \mu = -2,0, \eta = 0,5$. El modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 10)$	46
4.1.	Izquierda: Una red EDO define un campo vectorial que transforma continuamente el estado. Los círculos representan lugares de evaluación. Derecha: Una red residual define una secuencia discreta de transformaciones finitas. Las imágenes fueron tomadas de un repositorio de GitHub del autor de [6].	50
5.1.	Resultados de la <i>grid search</i> para diferentes configuraciones de la NeuralODE. El eje x representa el tamaño de las capas, el eje y representa la cantidad de capas, y el color indica el MSE.	63
5.2.	Resultados de la <i>grid search</i> para diferentes configuraciones de la NeuralODE. El eje x representa la profundidad o cantidad de capas, el eje y representa el error (MSE) y el tamaño de cada capa está indicado por el color.	64
5.3.	Comparación de las predicciones de la NeuralODE con los valores reales del sistema de ecuaciones que tiene condiciones iniciales completamente nuevas y aleatorias. Como podemos ver, el modelo se ajusta fielmente a la solución real.	65
5.4.	Comparación de las predicciones de la RNN junto con los valores reales y los valores pronosticados por la NeuralODE. El gráfico muestra la evolución de las variables T y E en el tiempo.	68

5.5. Comparación de las predicciones de la RNN junto con los valores reales con retardo y los valores pronosticados por la NeuralODE. El gráfico muestra la evolución de las variables T y E en el tiempo.	72
--	----

Índice de tablas

2.1. La existencia y la estabilidad de todos los equilibrios factibles del modelo (2.2) sin retardo temporal. Aquí, IE significa inestable, GAE-1 globalmente asintóticamente estable en el primer cuadrante del plano $T-E$, GAE-2 globalmente asintóticamente estable en el conjunto $\{(T, E) : T \geq 0, E > 0\}$, LAE localmente asintóticamente estable, NS nodo-silla.	23
3.1. Resumen de los resultados de las simulaciones basadas en los resultados documentados en la tabla (2.1).	42

Resumen

En este trabajo de grado, se investiga la eficacia de las ecuaciones diferenciales neuronales (NDEs) aplicadas a un modelo de crecimiento tumoral con retardo. Se analizan tanto modelos con ecuaciones diferenciales tradicionales como aquellos incorporando retardos temporales, destacando la importancia de estos últimos en la representación realista del crecimiento tumoral y la respuesta inmunológica.

El estudio se divide en varias secciones: una introducción a las ecuaciones diferenciales neuronales y su aplicación en modelización de sistemas dinámicos complejos; un análisis dinámico y de estabilidad del modelo tumoral sin retardo; y un análisis similar para el modelo con retardo temporal. Se implementan simulaciones numéricas utilizando Python, comparando resultados entre NDEs y redes neuronales recurrentes (RNNs), evaluando su precisión mediante el error cuadrático medio (MSE).

Los resultados indican que las NDEs superan a las RNNs en precisión y capacidad de generalización en modelos sin retardo. Sin embargo, para modelos con retardo, aunque las NDEs presentan mejores resultados que las RNNs, no alcanzan la precisión esperada. Finalmente, se concluye con una discusión sobre las ventajas y limitaciones de las NDEs en la modelización de sistemas biológicos complejos.

Este trabajo contribuye significativamente al campo de las matemáticas aplicadas y la biología, proporcionando una evaluación exhaustiva de nuevas metodologías para el modelado de dinámicas tumorales y abriendo vías para futuras investigaciones en el área.

Capítulo 1

Introducción

En el ámbito de las ecuaciones diferenciales, se despliega un campo de estudio emocionante que fusiona las ventajas de las redes neuronales con la modelización de sistemas dinámicos. Este fenómeno cobra una importancia especial en aplicaciones científicas e ingenieriles, donde la predicción y el análisis de sistemas complejos basados en datos temporales desempeñan un papel crucial. Las ecuaciones diferenciales neuronales ofrecen una solución elegante y potente para este tipo de desafíos, y dado su reciente desarrollo, un campo en el que todo ha ocurrido en los últimos cinco años, se encuentran innumerables desafíos y oportunidades de aporte significativo [6, 22].

A diferencia de los enfoques basados en la física y los modelos de redes neuronales convencionales, las ecuaciones diferenciales neuronales (NDEs por sus siglas en inglés) eliminan la necesidad de suposiciones funcionales previas, como la elección de formas específicas de funciones de transferencia o la necesidad de definir explícitamente la estructura de los datos subyacentes, simplificando así el proceso de modelado. Además, su flexibilidad en el manejo de series temporales con muestreo irregular e incompleto las convierte en herramientas altamente efectivas, como bien comprobó [6] al compararlas con las redes neuronales tradicionales.

En muchas aplicaciones, se asume que un sistema dependiente del tiempo sigue una propiedad conocida como Markovianidad, lo que implica que los futuros estados del sistema se determinan completamente por el estado actual y no se ven influenciados por eventos pasados. En tales casos, es posible describir eficazmente el sistema mediante ecuaciones diferenciales, ordinarias o parciales [10, 36]. Sin embargo, esta suposición de Markovianidad a menudo representa solo una primera aproximación a la realidad, ya que, en situaciones más realistas, los estados pasados del sistema también desempeñan un papel crucial. Para abordar estas situaciones, se ha desarrollado ampliamente la teoría de las ecuaciones diferenciales con

retardo. Una generalización de la teoría de ecuaciones diferenciales ordinarias, la cual, ha sido dotada un andamiaje teórico que ha llevado un paso más allá, la teoría clásica de sistemas dinámicos [1, 9, 38]. Sin embargo, y a pesar de involucrar casi todas las ramas de la matemática, este andamiaje solo nos brinda teoremas de existencia y algunos métodos explícitos, pero de aplicación local, así que para la obtención de soluciones generales de carácter global y la visualización de las mismas, aún seguimos dependiendo de métodos numéricos, el ambiente perfecto para la inclusión de las NDEs.

Este trabajo tiene como objetivo principal evaluar la eficacia de las ecuaciones diferenciales neuronales al incorporarlas en un modelo de crecimiento tumoral y contrastar sus resultados con los métodos tradicionales para solucionar este tipo de modelos, para esto tomaremos como caso de estudio un modelo de alta complejidad matemática, un modelo de crecimiento tumoral altamente no lineal, considerado con y sin retardo, lo que nos permitirá contrastar los resultados tradicionales obtenidos a partir de la teoría clásica de sistemas dinámicos para solucionar este tipo de modelos, con la información obtenida a partir de una NDEs construidas y adaptada a este modelo.

Para esto nos planteamos tres objetivos específicos en primer lugar, estudiar y documentar la teoría básica de las ecuaciones diferenciales neuronales, En segundo lugar, aplicar la teoría de las ecuaciones diferenciales neuronales a un modelo que describe el crecimiento de un tumor y por último evaluar la eficacia de las ecuaciones diferenciales neuronales a partir de los resultados obtenidos.

Las contribuciones de este trabajo son múltiples, de gran relevancia y con aportes completamente originales, en el campo de las matemáticas aplicadas a la biología y en el campo del desarrollo de herramientas computacionales. A continuación presentamos de manera desglosada los principales aportes que presenta este documento:

1. En el artículo de Jianquan Li *et al.* [30] del año 2021 presentan un modelo de crecimiento tumoral incorporando un retraso en la acción del tumor. En la sección 2.1 presentamos una revisión exhaustiva de como se llega al modelo descrito en [30] a partir de los trabajos previos de [2, 4, 11, 14, 20, 27, 32, 34].
2. Para el modelo que presentan Jianquan Li *et al.* calculan las soluciones de equilibrios y determinan su estabilidad local y global para el modelo de crecimiento tumoral con retardo. Sin embargo, no muestran ningún cálculo específico para el modelo sin retardo, en la sección 2.2 nosotros desarrollamos de manera altamente detallada los cálculos para el modelo sin retardo, lo cual es un aporte completamente original, estos cálculos son necesarios para el desarrollo posterior de las soluciones de equilibrios y estabilidad del modelo con retardo que desarrollamos de manera exhaustiva en la

sección 2.3.

3. Jianquan Li *et al.* también realizaron una serie de implementaciones para analizar los efectos del retardo en el modelo de crecimiento tumoral, en este trabajo replicamos sus resultados usando una implementación en python y adicionalmente analizamos nuevos resultados variando los parámetros del modelo.
4. A través de la implementación de estos modelos en Python con NDEs y el análisis comparativo de los resultados obtenidos, se busca evidenciar las ventajas y limitaciones de las NDEs en la modelización de sistemas dinámicos complejos.
5. Se ofrece una implementación de NeuralODEs adaptadas para un sistema que describe el crecimiento de un tumor sin retardo, demostrando su superioridad en términos de precisión y capacidad de generalización en comparación con las redes neuronales recurrentes (RNNs) tradicionales.
6. Los resultados indican que las NeuralODEs son capaces de capturar dinámicas complejas con mayor exactitud, lo que se refleja en menores valores de error cuadrático medio (MSE). En segundo lugar, utilizamos la implementación de las NeuralODEs para el sistema que describe el crecimiento de un tumor con retardo.
7. Sin embargo, los resultados obtenidos no eran los esperados en términos de precisión; no obstante, en comparación con las RNNs tradicionales, siguen reflejando mejores resultados al comparar los MSE.

Como resultado de esta investigación, actualmente estamos trabajando en el desarrollo del documento que nos permita registrar ante las autoridades correspondientes la patente de la herramienta que desarrollamos para aplicar la teoría de las NeuralODEs al sistema de crecimiento tumoral, en la redacción de un artículo de divulgación, en el cual mostraremos los beneficios y desventajas de aplicar las NDEs en sistemas de alta no linealidad y estamos trabajando en la aplicación de las NDEs en un modelo de crecimiento tumoral donde la función de estimulación de las células tumorales no es la Michaelis-Menten, sino una función de tipo logístico que la generaliza.

Capítulo 2

Un modelo de crecimiento tumoral en ecuaciones diferenciales con y sin retraso

2.1. Antecedentes biológicos y origen del modelo.

Un tumor es un crecimiento anormal de tejido. Cuando un tejido desconocido aparece en el cuerpo, el sistema inmune trata de identificarlo y, si es posible, tratar de eliminarlo. La respuesta del sistema inmune consiste en dos respuestas diferentes que interactúan, la respuesta celular y la respuesta humoral. La respuesta celular la llevan a cabo los linfocitos T . La respuesta humoral está relacionada con la otra clase de células llamadas linfocitos B . La respuesta inmunitaria comienza cuando las células tumorales son reconocidas como no propias. Entonces las células tumorales son atrapadas por los macrófagos. Los macrófagos absorben las células tumorales, se las comen y liberan una serie de citocinas que activan las células ayudantes T (es decir, una subpoblación de linfocitos T) que coordinan el contraataque. Estos linfocitos auxiliares no pueden destruir las células tumorales, pero envían señales bioquímicas urgentes a un tipo especial de linfocitos T denominados asesinos naturales (NK por sus siglas en inglés). Los linfocitos T comienzan a multiplicarse y a liberar otras citocinas que estimulan aún más a más linfocitos T , linfocitos B y células NK. A medida que aumenta el número de células B , las células ayudantes T envían una señal para iniciar el proceso de producción de anticuerpos. Los anticuerpos circulan por la sangre y se adhieren a las células tumorales, lo que implica que son engullidas más rápidamente por los macrófagos o eliminadas por las células asesinas naturales. Como todas las células T , las células NK están entrenadas para reconocer un tipo específico de célula infectada o cancerosa. Las células NK son letales. Constituyen una línea crítica de la defensa. La formación de un tumor ocurre a menudo debido a problemas en el sistema inmune [11]. La

dinámica de la respuesta inmunitaria antitumoral *en vivo* es complicada y no se conoce bien, por lo que numerosos modelos matemáticos se han desarrollado acerca del crecimiento de un tumor. En [27], Kuznetsov et al. proponen un modelo matemático que describe la interacción entre las células efectoras y las células tumorales

$$\begin{cases} \frac{dT}{dt} = rT \left(1 - \frac{T}{K}\right) - nET, \\ \frac{dE}{dt} = \sigma + \frac{pET}{g+T} - mET - \eta E. \end{cases} \quad (2.1)$$

Donde $T = T(t)$ y $E = E(t)$ son las concentraciones de células tumorales y células efectoras con respecto al tiempo t , respectivamente, r es el coeficiente de crecimiento máximo de las células tumorales, K es la capacidad de carga del entorno biológico para las células tumorales, n es el coeficiente de la tasa a la que las células tumorales son dañadas por las células efectoras, σ es la tasa normal (no aumentada por la presencia del tumor) de entrada de células efectoras maduras en la zona tumoral, m es el coeficiente de la tasa de disminución de las células efectoras neutralizadas por las células tumorales, y η es el coeficiente de la tasa de eliminación natural de las células efectoras.

En el modelo (2.1), se supone que, además de la afluencia natural, las células efectoras también pueden ser reclutadas por estimulación de las células tumorales. La tasa correspondiente se describe mediante la función de Michaelis-Menten:

$$\frac{pET}{g+T}$$

en la que p es el coeficiente máximo y g es la constante asociada. Siguiendo esta dirección, Galach, en 2003 [14], simplificó el modelo (2.1) sustituyendo la función de Michaelis-Menten por una forma bilineal pET de la ley de acción de masa simple, con esto el modelo (2.1) se reduce a:

$$\begin{cases} \frac{dT}{dt} = rT \left(1 - \frac{T}{K}\right) - nET, \\ \frac{dE}{dt} = \sigma + pET - mET - \eta E = \sigma + \mu TE - \eta E, \end{cases} \quad (2.2)$$

donde $\mu = p - m$.

Los dos términos centrales de la ecuación para E en el modelo (2.1) reflejan los efectos de la estimulación de las células tumorales y la neutralización en el crecimiento de las células efectoras, respectivamente. Entonces, el término:

$$\frac{pET}{g+T} - mET$$

en el modelo (2.1) puede denominarse efecto de las células tumorales sobre las células efectoras. Por lo tanto, nos referimos al parámetro μ en el modelo (2.2) como el coeficiente de acción de las células tumorales sobre las células efectoras. Concretamente, $\mu < 0$ significa que en el proceso de interacción entre las células

efectoras y las células tumorales, la neutralización de las células efectoras por parte de las células tumorales supera a la estimulación, es decir, el efecto inhibitor de las células tumorales es dominante. Por el contrario, el efecto estimulante es dominante cuando $\mu > 0$. El caso $\mu = 0$ es trivial y se omitirá. Correspondiente al modelo (2.1) Khajanchi y Banerjee [20] describen la dinámica de las interacciones tumor-inmune con un retardo temporal como:

$$\begin{cases} \frac{dT(t)}{dt} = rT(t) \left(1 - \frac{T(t)}{K}\right) - nE(t)T(t), \\ \frac{dE(t)}{dt} = \sigma + \frac{pE(t-\tau)T(t-\tau)}{g+T(t-\tau)} - mE(t)T(t) - \eta E(t), \end{cases} \quad (2.3)$$

donde el retardo τ refleja un lapso de tiempo entre la acumulación de las células efectoras alrededor de las células tumorales y la interacción con las mismas células tumorales. Además, Galach [14] incluyó en el modelo (2.2) el tiempo que desarrolla una respuesta adecuada de las células efectoras tras la aparición de células tumorales, lo que le condujo al siguiente modelo con retardo:

$$\begin{cases} \frac{dT(t)}{dt} = rT(t) \left(1 - \frac{T(t)}{K}\right) - nE(t)T(t), \\ \frac{dE(t)}{dt} = \sigma + \mu T(t-\tau)E(t-\tau) - \eta E(t). \end{cases} \quad (2.4)$$

En este caso, τ representa el retardo en el tiempo de respuesta de las células efectoras tras la aparición de células tumorales. Luego, Bi y Xiao [2] introdujeron otro retardo (τ') en el modelo (2.2) para obtener el siguiente sistema:

$$\begin{cases} \frac{dT(t)}{dt} = rT(t) \left(1 - \frac{T(t)}{K}\right) - nE(t-\tau')T(t-\tau'), \\ \frac{dE(t)}{dt} = \sigma + \mu T(t-\tau)E(t-\tau) - \eta E(t), \end{cases} \quad (2.5)$$

Similares a los modelos (2.3), (2.4) y (2.5), algunos retardos de tiempo discreto se incluyen en otros modelos tumorales [2, 4]. Se señaló en [14] que, cuando $\mu \geq 0$, las soluciones del modelo (2.4) son no negativas. Sin embargo, existen condiciones iniciales no negativas, tales que las soluciones $E(t)$ y el modelo (2.4) se hacen negativos en un intervalo de tiempo finito si $\mu < 0$, lo cual es inconsistente con la realidad. Para el modelo (2.5) también existe una deficiencia similar.

Podemos notar que entre los efectos del tumor sobre el sistema inmune, está la estimulación e inhibición del tumor para las células efectoras, y los dos efectos son realizados por los antígenos liberados por las células tumorales. En primer lugar, las células que presentan antígenos despiertan una respuesta inmunitaria, determinados linfocitos, como las células T, reconocen los antígenos generados por las células tumorales y activan las células efectoras. En segundo lugar, la aparición de antígenos en el cuerpo tiene cierto efecto nocivo sobre las células efectoras; es decir, inhiben el crecimiento de las células efectoras.

Como se resalta en [19], la fuerza de la respuesta inmunitaria puede estar regida por la dosis de antígeno, la localización y las señales coestimuladoras, que refuerza la señal del antígeno, en este y otros trabajos similares como [32, 34], se muestra que la respuesta inmunitaria al cáncer está determinada por el equilibrio entre la antigenicidad del tumor y el microambiente de los tejidos cancerosos. Así, en muchos casos, se necesita tiempo para que el tumor progrese para actuar sobre el sistema inmunitario. Más concretamente, si dejamos que el período de tiempo que tardan en progresar las células tumorales sea τ , las células efectoras en el momento t son esencialmente afectadas por las células tumorales originadas en el momento $t - \tau$. Así en [30] finalmente se propone un modelo que a diferencia de (2.3), (2.4) y (2.5), en los que se supone que la acción de las células tumorales sobre las efectoras es instantánea, pero la aparición de los efectos se retrasa, este nuevo modelo expresa el efecto de las células tumorales sobre las efectoras mediante el término $\mu T(t - \tau)E(t)$, obteniendo así

$$\begin{cases} \frac{dT(t)}{dt} = rT(t) \left(1 - \frac{T(t)}{K}\right) - nE(t)T(t), \\ \frac{dE(t)}{dt} = \sigma + \mu T(t - \tau)E(t) - \eta E(t). \end{cases} \quad (2.6)$$

Claramente, las soluciones del modelo (2.6) con condiciones iniciales no negativas se mantienen no negativas para $t > 0$. Si se ignora el retardo de tiempo cuando el tumor actúa sobre el sistema inmune, es decir, $\tau = 0$, el modelo (2.6) es igual que el modelo (2.2), y no presenta dinámicas más complejas como la existencia de soluciones periódicas producidas por bifurcación de Hopf [31]. A diferencia del modelo (2.2), la inclusión del retardo τ también cambia la dependencia de la dinámica del modelo (2.6) de otros parámetros.

Dado que el significado biológico de τ aquí es diferente que en el modelo (2.4), llamamos a τ el tiempo de retardo de la acción del tumor. En consecuencia, el modelo (2.6) está sujeto a las condiciones iniciales

$$\begin{aligned} T(\theta) &= \psi_1(\theta), \quad E(\theta) = \psi_2(\theta), \\ \psi_i(\theta) &\geq 0, \quad \theta \in [-\tau, 0], \quad \psi_i(0) > 0, \quad i = 1, 2, \end{aligned} \quad (2.7)$$

donde $(\psi_1(\theta), \psi_2(\theta)) \in \mathbb{C}([-\tau, 0], \mathbb{R}_0^2)$, el espacio de Banach de funciones continuas que mapean el intervalo $[-\tau, 0]$ en \mathbb{R}_{+0} con norma $\|\psi\| = \sup_{-\tau \leq \theta \leq 0} |\psi_i(\theta)|$, $i = 1, 2$, y $\mathbb{R}_{+0}^2 = \{(x_1, x_2) \mid x_i \geq 0, i = 1, 2\}$.

En este trabajo queremos enfocarnos en el modelo (2.6) y aplicarle a la teoría de las ecuaciones diferenciales neuronales, para esto tenemos que comprender el comportamiento del modelo, con un análisis de estabilidad que nos permitirá predecir cómo el sistema responderá a perturbaciones y cambios en las condiciones iniciales. Además, en modelos matemáticos, especialmente en los que representan fenómenos físicos, biológicos o económicos, la estabilidad proporciona una validación de que el modelo es realista y

útil. Inicialmente, vamos a estudiar el sistema (2.2) que correspondería a nuestro modelo sin retardo. Luego, realizaremos un análisis similar para el modelo (2.6), observaremos el impacto del retardo con simulaciones numéricas y con estas herramientas entraremos en el siguiente capítulo enfocado en las ecuaciones diferenciales neuronales.

2.2. Análisis dinámico del modelo sin retardo

Recordemos nuevamente el modelo (2.2) sin retardo de tiempo,

$$\begin{cases} \frac{dT}{dt} = rT \left(1 - \frac{T}{K}\right) - nET, \\ \frac{dE}{dt} = \sigma + \mu TE - \eta E, \end{cases} \quad (2.2)$$

para el cual se demuestran las propiedades básicas a continuación (los preliminares para entender esta sección se encuentran en el Apéndice A):

Lema 2.1. *Para toda condición inicial no negativa (T_0, E_0) , existe una solución única no negativa $T(t), E(t)$ de (2.2) para todo $t > 0$*

Demostración. Puesto que el lado derecho de (2.2) es un polinomio, existe una solución local única para cualquier condición inicial.

Para empezar, la ecuación diferencial para $T(t)$ es:

$$\frac{dT}{dt} = rT \left(1 - \frac{T}{K}\right) - nET$$

Reescribimos la ecuación para hacerla más manejable:

$$\frac{dT}{dt} = T \left(r \left(1 - \frac{T}{K}\right) - nE \right)$$

Para resolver esta ecuación, podemos usar el método de separación de variables. Dividimos ambos lados por T :

$$\frac{1}{T} \frac{dT}{dt} = r \left(1 - \frac{T}{K}\right) - nE$$

Integramos ambos lados con respecto al tiempo t :

$$\int \frac{1}{T} dT = \int \left(r \left(1 - \frac{T}{K}\right) - nE \right) dt$$

Esto nos da:

$$\ln(T) = \int_0^t \left(r \left(1 - \frac{T(s)}{K} \right) - nE(s) \right) ds + C$$

Donde C es una constante de integración. Luego, exponenciamos ambos lados para despejar $T(t)$:

$$T(t) = e^{\int_0^t \left(r \left(1 - \frac{T(s)}{K} \right) - nE(s) \right) ds + C}$$

Si consideramos la condición inicial $T(0) = T_0$, obtenemos que:

$$T_0 = e^C$$

Por lo tanto, $e^C = T_0$, y finalmente tenemos:

$$T(t) = T_0 e^{\int_0^t \left(r \left(1 - \frac{T(s)}{K} \right) - nE(s) \right) ds}$$

Por otro lado, la ecuación diferencial para $E(t)$ es:

$$\frac{dE}{dt} = \sigma + \mu T E - \eta E$$

Reescribimos la ecuación para hacerla más manejable:

$$\frac{dE}{dt} = \sigma + (\mu T - \eta) E$$

Para resolver esta ecuación, podemos usar el método de separación de variables. Dividimos ambos lados por E :

$$\frac{1}{E} \frac{dE}{dt} = \frac{\sigma}{E} + \mu T - \eta$$

Integramos ambos lados con respecto al tiempo t :

$$\int \frac{1}{E} dE = \int \left(\frac{\sigma}{E} + \mu T - \eta \right) dt$$

Esto nos da:

$$\ln(E) \geq \int_0^t (\mu T(s) - \eta) ds + C$$

Donde C es una constante de integración. Luego, exponenciamos ambos lados para despejar $E(t)$:

$$E(t) \geq e^{\int_0^t (\mu T(s) - \eta) ds + C}$$

Si consideramos la condición inicial $E(0) = E_0$, obtenemos que:

$$E_0 = e^C$$

Por lo tanto, $e^C = E_0$, y finalmente tenemos:

$$E(t) \geq E_0 e^{\int_0^t (\mu T(s) - \eta) ds}$$

Por lo que podemos ver que para todo $t > 0$

$$T(t) = T_0 e^{\int_0^t \left(r \left(1 - \frac{T(s)}{K} \right) - n E(s) \right) ds},$$

$$E(t) \geq E_0 e^{\int_0^t (\mu T(s) - \eta) ds}.$$

Así, si $T_0, E_0 \geq 0$, entonces $T(t)$ y $E(t)$ siguen siendo no negativos para todo $t > 0$. Dado que $T(t)$ y $E(t)$ son no negativos, tenemos

$$\frac{dT}{dt} \leq rT \left(1 - \frac{T}{K} \right)$$

y entonces esto implica que $T(t)$ no puede crecer indefinidamente y está acotado por un T_{\max} es decir:

$$T(t) \leq \max(T_0, K) = T_{\max}.$$

Usando T_{\max} , podemos estimar para la segunda ecuación:

$$\frac{dE}{dt} = \sigma + \gamma E$$

lo que implica que $\gamma = \mu T_{\max} - \eta$ siempre que $\mu \geq 0$ y luego considerando la ecuación acotada:

$$E(t) \leq E_0 e^{\gamma t} + \sigma e^{\gamma t} \int_0^t e^{-\gamma s} ds,$$

donde $\gamma = -\eta$ si $\mu < 0$ por lo que tenemos que:

$$\gamma = \begin{cases} \mu T_{\max} - \eta & \text{si } \mu \geq 0, \\ -\eta & \text{si } \mu < 0. \end{cases}$$

Esto implica que para cualquier momento de tiempo finito, $T(t)$ y $E(t)$ están acotados, y esta es una condición suficiente para la existencia de soluciones para todo $t > 0$.

□

Ahora estudiamos el comportamiento asintótico del modelo, para el cual existen hasta tres estados estacionarios. El estado estacionario $P_0 = P_0\left(0, \frac{\sigma}{\eta}\right)$ que denota el equilibrio sin tumor y siempre existe. Los demás equilibrios positivos del modelo (2.2) pueden encontrarse resolviendo directamente el sistema

$$\begin{cases} r\left(1 - \frac{T}{K}\right) = nE, \\ \sigma = (\eta - \mu T)E, \end{cases} \quad (2.8)$$

que se obtiene dejando que los lados derechos de (2.2) sean iguales a ceros. Resolvemos el sistema de ecuaciones por sustitución para T y E que nos da como resultado

$$\begin{aligned} \frac{\mu r}{nk} T^2 - \left(\frac{\eta r}{nk} + \frac{\mu r}{n}\right) T + \frac{\eta r}{n} - \sigma, \\ n\mu k E^2 - (r(\mu k + \eta))E + \sigma r. \end{aligned}$$

Luego, resolvemos por fórmula cuadrática y optemos los equilibrios positivos $P^* = P^*(T^*, E^*)$, $P_* = P_*(T_*, E_*)$ y $P^* = P^*(T^*, E^*)$ donde

$$\begin{aligned} T^* &= \frac{K}{2\mu} \left[\left(\mu + \frac{\eta}{K} \right) - \sqrt{\Delta} \right], & E^* &= \frac{r}{2n\mu} \left[\left(\mu - \frac{\eta}{K} \right) + \sqrt{\Delta} \right], \\ T_* &= \frac{K}{2\mu} \left[\left(\mu + \frac{\eta}{K} \right) + \sqrt{\Delta} \right], & E_* &= \frac{r}{2n\mu} \left[\left(\mu - \frac{\eta}{K} \right) - \sqrt{\Delta} \right], & \Delta &= \left(\mu - \frac{\eta}{K} \right)^2 + \frac{4n\sigma\mu}{Kr}, \\ T_*^* &= \frac{K}{2\mu} \left(\mu + \frac{\eta}{K} \right), & E_*^* &= \frac{r}{2n\mu} \left(\mu - \frac{\eta}{K} \right). \end{aligned} \quad (2.9)$$

2.2.1. Estabilidad para el estado estacionario P_0

Ahora para el análisis de la estabilidad local vamos a calcular la matriz jacobiana del sistema (2.2) dada por

$$\begin{bmatrix} r - 2\frac{rT}{K} - nE & -nT \\ \mu E & \mu T - \eta \end{bmatrix}. \quad (2.10)$$

De donde el polinomio característico de (2.10) asociado al estado estacionario P_0 es,

$$\lambda^2 + \lambda \left(-r - \frac{n\sigma}{\eta} \right) + (r\eta + n\sigma). \quad (2.11)$$

De la ecuación (2.11) podemos hallar los autovalores de (2.10) por lo que tenemos:

$$\begin{aligned} \lambda_1 &= \frac{r - \frac{n\sigma}{\eta} + \sqrt{\left(-r + \frac{n\sigma}{\eta} \right)^2 - 4(r\eta + n\sigma)}}{2}, \\ \lambda_2 &= \frac{r - \frac{n\sigma}{\eta} - \sqrt{\left(-r + \frac{n\sigma}{\eta} \right)^2 - 4(r\eta + n\sigma)}}{2}. \end{aligned} \quad (2.12)$$

Por lo tanto, podemos decir que el estado estacionario P_0 es localmente asintóticamente estable cuando $n\sigma > r\eta$ ya que esto hace que los autovalores tomen partes reales negativas, e inestable cuando $n\sigma < r\eta$.

2.2.2. Estabilidad para el estado estacionario P^*

Primero tenemos que analizar bajo que condiciones P^* existe y es positivo, recordemos de las condiciones (2.9) como está definido P^* se tiene

$$\begin{aligned} T^* &= \frac{K}{2\mu} \left(\left(\mu + \frac{\eta}{K} \right) - \sqrt{\Delta} \right), \\ E^* &= \frac{r}{2n\mu} \left(\left(\mu - \frac{\eta}{K} \right) + \sqrt{\Delta} \right), \end{aligned} \quad \Delta = \left(\mu - \frac{\eta}{K} \right)^2 + \frac{4n\sigma\mu}{Kr}. \quad (2.13)$$

Para que T^* sea no negativo, debe cumplirse que:

$$\left(\mu + \frac{\eta}{K} \right) - \sqrt{\Delta} \geq 0.$$

$$\left(\mu + \frac{\eta}{K} \right) \geq \sqrt{\Delta}.$$

Elevando al cuadrado ambos lados:

$$\left(\mu + \frac{\eta}{K} \right)^2 \geq \left(\mu - \frac{\eta}{K} \right)^2 + \frac{4n\sigma\mu}{Kr}.$$

Expandimos y simplificamos:

$$\mu^2 + 2\mu\frac{\eta}{K} + \left(\frac{\eta}{K} \right)^2 \geq \mu^2 - 2\mu\frac{\eta}{K} + \left(\frac{\eta}{K} \right)^2 + \frac{4n\sigma\mu}{Kr}.$$

Cancelamos términos comunes y reorganizamos:

$$4\mu \frac{\eta}{K} \geq \frac{4n\sigma\mu}{Kr},$$

$$4\mu\eta \geq \frac{4n\sigma\mu}{r}.$$

Dividiendo por 4 y por μ (asumiendo $\mu \neq 0$):

$$\eta r \geq n\sigma.$$

Para que E^* sea no negativo, debe cumplirse que:

$$\left(\mu - \frac{\eta}{K}\right) + \sqrt{\Delta} \geq 0.$$

Recordemos que Δ está definida como:

$$\Delta = \left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}.$$

Sustituimos Δ :

$$\left(\mu - \frac{\eta}{K}\right) + \sqrt{\left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}} \geq 0.$$

Ambas condiciones $T^* \geq 0$ y $E^* \geq 0$ deben cumplirse simultáneamente para que P^* sea no negativo. Bajo estas condiciones podemos plantear ciertos casos en donde garantizamos la existencia y no negatividad de P^*

1. $\mu > 0, n\sigma < r\eta$:

Es claro que estas condiciones garantizan la existencia y no negatividad de P^*

2. $\mu < 0, \eta + \mu K \geq 0, n\sigma < r\eta$:

La condición para E^* es:

$$\left(\mu - \frac{\eta}{K}\right) + \sqrt{\left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}} \geq 0.$$

Sí $\eta + \mu K \geq 0$:

- Esto implica que $\eta \geq -\mu K$.

- Dado que $\mu < 0$, la condición $(\mu - \frac{\eta}{K}) \geq 0$ no se puede cumplir porque μ es negativo y $\frac{\eta}{K}$ sería positivo en este caso.
- Sin embargo, dado que $\eta \geq -\mu K$, el término $\sqrt{\Delta}$ puede compensar el término negativo $(\mu - \frac{\eta}{K})$.

3. $\mu < 0, \eta + \mu K < 0, n\sigma \leq r\eta$ La condición para E^* es:

$$\left(\mu - \frac{\eta}{K}\right) + \sqrt{\left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}} \geq 0.$$

Sí $\eta + \mu K < 0$:

- Esto implica que $\eta < -\mu K$.
 - Dado que $\mu < 0$, el término $(\mu - \frac{\eta}{K})$ será negativo, si $\frac{\eta}{K} > \mu$.
 - En este caso, la condición $\eta + \mu K < 0$ asegura que el término $\sqrt{\Delta}$ pueda compensar el término negativo.
4. $\mu < 0, \eta + \mu K < 0, r\eta < n\sigma < -\frac{rK}{4\mu}(\mu - \frac{\eta}{K})^2$ Para que P^* exista y sea no negativo, necesitamos que ambas condiciones se cumplan simultáneamente:

$$\sqrt{\Delta} \geq \left|\mu + \frac{\eta}{K}\right| \quad \text{y} \quad \sqrt{\Delta} \geq \left|-\mu + \frac{\eta}{K}\right|.$$

Dado que $\Delta = (\mu - \frac{\eta}{K})^2 + \frac{4n\sigma\mu}{Kr}$, analizamos bajo las condiciones del caso C4:

$$r\eta < n\sigma < -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2.$$

Esto asegura que Δ es suficientemente grande para cumplir ambas condiciones. Específicamente:

$$\Delta = \left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr},$$

dado que $\mu < 0$, y $n\sigma > r\eta$,

$$\Delta = \left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr},$$

y como $n\sigma < -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$,

$$\Delta = \left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}.$$

Por lo tanto, Δ es positivo y suficientemente grande para que:

$$\sqrt{\Delta} \geq \left| \mu + \frac{\eta}{K} \right| \quad \text{y} \quad \sqrt{\Delta} \geq \left| -\mu + \frac{\eta}{K} \right|,$$

lo cual asegura que $T^* \geq 0$ y $E^* \geq 0$.

Ahora vamos a sustituir P^* en (2.10)

$$J = \begin{bmatrix} r - 2\frac{rT^*}{K} - nE^* & -nT^* \\ \mu E^* & \mu T^* - \eta \end{bmatrix}. \quad (2.14)$$

La traza ($\text{tr}(J)$) y el determinante ($\det(J)$) del jacobiano son:

$$\text{tr}(J) = r \left(1 - \frac{2T^*}{K} \right) - nE^* + \mu T^* - \eta, \quad (2.15)$$

$$\det(J) = \left[r \left(1 - \frac{2T^*}{K} \right) - nE^* \right] [\mu T^* - \eta] - (-nT^*)(\mu E^*). \quad (2.16)$$

Sustituimos T^* y E^* y simplificamos

$$\text{tr}(J) = \frac{r\sqrt{\Delta}}{2\mu} - \frac{K\sqrt{\Delta}}{2} + \frac{K\mu}{2} - \frac{\eta}{2} - \frac{r}{2} - \frac{r\eta}{2K\mu}, \quad (2.17)$$

$$\begin{aligned} \det(J) &= \left(\frac{r\sqrt{\Delta}}{2\mu} - \frac{r\eta}{2K\mu} - \frac{r}{2} \right) \left(\frac{K\mu}{2} - \frac{\eta}{2} - \frac{K\sqrt{\Delta}}{2} \right) \\ &+ \left(\frac{rK}{4\mu^2} \left(\mu^2 - \left(\frac{\eta}{K} \right)^2 - (\sqrt{\Delta})^2 + \sqrt{\Delta} \cdot \mu + \sqrt{\Delta} \cdot \frac{\eta}{K} \right) \right). \end{aligned} \quad (2.18)$$

Ahora, Para que el punto de equilibrio $P^* = (T^*, E^*)$ sea localmente asintóticamente estable, necesitamos que:

- La traza del Jacobiano sea negativa ($\text{tr}(J) < 0$).
- El determinante del Jacobiano sea positivo ($\det(J) > 0$).

Dados anteriormente los casos donde P^* existe y es no negativo, se puede comprobar que para todas las condiciones esto se cumple. Por lo que P^* en caso de existir es estable.

2.2.3. Estabilidad para el estado estacionario P_*

En esta caso para que T_* y E_* sean no negativos, se deben cumplir las siguientes condiciones:

- Suponiendo $\mu > 0$

Para T_* no negativo:

$$\mu + \frac{\eta}{K} + \sqrt{\Delta} \geq 0.$$

Para E_* no negativo:

$$\mu - \frac{\eta}{K} - \sqrt{\Delta} \geq 0.$$

Ambas condiciones deben cumplirse simultáneamente:

$$\mu + \frac{\eta}{K} + \sqrt{\Delta} \geq 0 \quad \text{y} \quad \mu - \frac{\eta}{K} - \sqrt{\Delta} \geq 0.$$

Sin embargo, estas dos condiciones son contradictorias, ya que si $\sqrt{\Delta} > 0$, una de las dos condiciones será siempre negativa. Por lo tanto, no existen condiciones bajo las cuales tanto T_* como E_* sean no negativos para $\mu > 0$.

- Suponiendo $\mu < 0$

Para T_* no negativo:

$$\mu + \frac{\eta}{K} + \sqrt{\Delta} \leq 0.$$

Para E_* no negativo:

$$\mu - \frac{\eta}{K} - \sqrt{\Delta} \leq 0.$$

Ambas condiciones deben cumplirse simultáneamente:

$$\mu + \frac{\eta}{K} + \sqrt{\Delta} \leq 0 \quad \text{y} \quad \mu - \frac{\eta}{K} - \sqrt{\Delta} \leq 0.$$

Para que ambas condiciones se cumplan simultáneamente:

$$-\left(\mu + \frac{\eta}{K}\right) \geq \sqrt{\Delta} \geq -\left(\mu - \frac{\eta}{K}\right).$$

Estas condiciones se alinean con las del caso C4:

$$\mu < 0, \quad \eta + \mu K < 0, \quad r\eta < n\sigma < -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2.$$

La traza del Jacobiano en el punto de equilibrio P_* es:

$$\text{tr}(J) = r \left(1 - \frac{2T_*}{K} \right) - nE_* + \mu T_* - \eta.$$

Reemplazando T_* y E_* y simplificando

$$\text{tr}(J) = \frac{K\mu}{2} - \frac{r}{2}.$$

Para que la traza del Jacobiano en P_* sea positiva, se debe cumplir:

$$\frac{K\mu}{2} - \frac{r}{2} > 0 \quad \Rightarrow \quad K\mu > r.$$

Dado que $\mu < 0$, esta condición implica:

$$K\mu - r > 0.$$

Bajo estas condiciones, el punto de equilibrio P_* existe y es no negativo, y la traza del Jacobiano es siempre positiva. Por lo tanto, P_* si existe es inestable.

2.2.4. Estabilidad para el estado estacionario P_*

En este caso vamos a demostrar que P_* es un punto silla. El punto de equilibrio P_* se define como

$$\begin{aligned} T_*^* &= \frac{K}{2\mu} \left(\mu + \frac{\eta}{K} \right), \\ E_*^* &= \frac{r}{2n\mu} \left(\mu - \frac{\eta}{K} \right). \end{aligned}$$

Para demostrar que P_* es un punto silla, necesitamos analizar la estabilidad de este punto de equilibrio de nuevo, vamos a utilizar la traza y el determinante del Jacobiano evaluado en P_* , para esto reemplacemos T_* y E_* en la matriz (2.10).

$$\begin{aligned}
 J_{11} &= r \left(1 - \frac{2T_*^*}{K} \right) - nE_*^* \\
 &= -\frac{r\eta}{2K\mu} - \frac{r}{2}.
 \end{aligned}$$

$$\begin{aligned}
 J_{12} &= -nT_*^* \\
 &= -\frac{nK}{2\mu} \left(\mu + \frac{\eta}{K} \right).
 \end{aligned}$$

$$\begin{aligned}
 J_{21} &= \mu E_*^* \\
 &= \frac{r}{2n} \left(\mu - \frac{\eta}{K} \right).
 \end{aligned}$$

$$\begin{aligned}
 J_{22} &= \mu T_*^* - \eta \\
 &= \frac{K\mu}{2} - \frac{\eta}{2}.
 \end{aligned}$$

La traza (tr) y el determinante (\det) del Jacobiano en P_* son

$$\begin{aligned}
 \text{tr}(J) &= -\frac{r\eta}{2K\mu} - \frac{r}{2} + \frac{K\mu}{2} - \frac{\eta}{2}, \\
 \det(J) &= 0.
 \end{aligned}$$

Para un punto de equilibrio P_* ser un punto silla, una de las raíces del polinomio característico del Jacobiano debe ser positiva y la otra negativa. Esto se demuestra analizando la traza y el determinante del Jacobiano en P_* . Si el determinante es negativo, y la traza es positiva, entonces el punto de equilibrio P_* es un punto silla. Con un análisis similar al del caso 2 para P_* sabemos que P_* existe y es no negativo si $\mu < 0$ y $\eta + \mu K < 0$ con lo que fácilmente podemos comprobar que $\text{tr}(J) > 0$ y $\det(J) = 0$, de donde se deduce que P_* es un punto silla.

2.2.5. Estabilidad global de los estados estacionarios

Hasta este punto se llevó a cabo el análisis de la estabilidad local de los cuatro estados estacionarios. En este apartado veremos la estabilidad global del modelo que se refiere a la estabilidad en todo el espacio de estados del sistema. Un punto de equilibrio es globalmente estable si cualquier trayectoria del sistema, sin importar el punto de partida, eventualmente se acerca y permanece cerca de este punto de equilibrio a diferencia de la estabilidad local donde garantizamos la estabilidad únicamente en cercanías de punto de equilibrio.

Para determinar la estabilidad global del punto de equilibrio $P_0 = \left(0, \frac{\sigma}{\eta} \right)$ analizaremos el compor-

tamiento del sistema de ecuaciones diferenciales en torno a P_0 y utilizaremos una función de Lyapunov (Sección A.3.1 del Apéndice A). Para esto, consideremos la función

$$V(T, E) = T + \frac{\eta}{2\sigma} \left(E - \frac{\sigma}{\eta}\right)^2,$$

esta función es positiva definida, ya que

$$V(T, E) \geq 0 \quad \text{y} \quad V(0, \frac{\sigma}{\eta}) = 0,$$

la derivada de V con respecto al tiempo t es

$$\frac{dV}{dt} = \frac{\partial V}{\partial T} \frac{dT}{dt} + \frac{\partial V}{\partial E} \frac{dE}{dt}.$$

Calculamos las derivadas parciales de V

$$\frac{\partial V}{\partial T} = 1,$$

$$\frac{\partial V}{\partial E} = \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right).$$

Entonces

$$\frac{dV}{dt} = \frac{dT}{dt} + \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right) \frac{dE}{dt}.$$

Sustituyendo las ecuaciones del sistema

$$\frac{dT}{dt} = rT \left(1 - \frac{T}{K}\right) - nET,$$

$$\frac{dE}{dt} = \sigma + \mu TE - \eta E,$$

obtenemos

$$\frac{dV}{dt} = \left[rT \left(1 - \frac{T}{K}\right) - nET \right] + \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right) (\sigma + \mu TE - \eta E).$$

Simplificamos la expresión

$$\frac{dV}{dt} = rT \left(1 - \frac{T}{K}\right) - nET + \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right) \sigma + \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right) \mu TE - \frac{\eta}{\sigma} \left(E - \frac{\sigma}{\eta}\right) \eta E.$$

$$= rT \left(1 - \frac{T}{K}\right) - nET + \eta E - \sigma + \frac{\eta \mu T E^2}{\sigma} - \frac{\eta^2 E}{\sigma} + \frac{\eta \sigma}{\sigma} - \eta E.$$

$$= rT \left(1 - \frac{T}{K}\right) - nET + \frac{\eta\mu TE^2}{\sigma} - \frac{\eta^2 E}{\sigma}.$$

Ahora, vamos a evaluar la derivada con base a dos casos específicos.

- $\mu > 0, \quad n\sigma \geq r\eta.$

Para $\mu > 0$

$$\frac{dV}{dt} = rT \left(1 - \frac{T}{K}\right) - nET + \frac{\eta\mu TE^2}{\sigma} - \frac{\eta^2 E}{\sigma}.$$

Dado que $\mu > 0$ y $n\sigma \geq r\eta$, podemos analizar los términos

- $rT \left(1 - \frac{T}{K}\right)$ es negativo o cero para $T > 0$.
- nET es negativo, ya que T y E son positivos.
- $\frac{\eta\mu TE^2}{\sigma}$ es positivo.
- $\frac{\eta^2 E}{\sigma}$ es negativo.

Como $n\sigma \geq r\eta$, esto implica que los términos negativos dominan, lo que lleva a $\frac{dV}{dt} \leq 0$.

- $\mu < 0, \quad \eta + \mu K < 0, \quad n\sigma > -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2.$

Para $\mu < 0$:

$$\frac{dV}{dt} = rT \left(1 - \frac{T}{K}\right) - nET + \frac{\eta\mu TE^2}{\sigma} - \frac{\eta^2 E}{\sigma}.$$

Dado que $\mu < 0$, analizamos los términos

- $rT \left(1 - \frac{T}{K}\right)$ es negativo o cero para $T > 0$.
- nET es negativo.
- $\frac{\eta\mu TE^2}{\sigma}$ es negativo.
- $\frac{\eta^2 E}{\sigma}$ es negativo.

Con todo lo anterior hemos probado que la función

$$V(T, E) = T + \frac{\eta}{2\sigma} \left(E - \frac{\sigma}{\eta}\right)^2$$

es una función de Lyapunov. Además, como en los dos casos analizados $\frac{dV}{dt} \leq 0$, y $V(T, E)$ es positiva definida. Esto implica que P_0 es globalmente asintóticamente estable bajo estas condiciones impuestas.

Ahora queremos demostrar la estabilidad global del punto de equilibrio $P^* = (T^*, E^*)$ en los primeros tres casos que analizamos, veremos el comportamiento del sistema de ecuaciones diferenciales en torno a

P^* y utilizaremos una función de Lyapunov.

Para demostrar la estabilidad global de P^* , consideramos una función de Lyapunov candidata. Una posible función de Lyapunov para este sistema es:

$$V(T, E) = \frac{(T - T^*)^2}{T^*} + \frac{(E - E^*)^2}{E^*}.$$

Esta función es positiva definida, ya que $V(T, E) \geq 0$ y $V(T^*, E^*) = 0$.

La derivada de V con respecto al tiempo t es

$$\frac{dV}{dt} = \frac{\partial V}{\partial T} \frac{dT}{dt} + \frac{\partial V}{\partial E} \frac{dE}{dt}.$$

Calculamos las derivadas parciales de V

$$\frac{\partial V}{\partial T} = \frac{2(T - T^*)}{T^*},$$

$$\frac{\partial V}{\partial E} = \frac{2(E - E^*)}{E^*}.$$

Entonces

$$\frac{dV}{dt} = \frac{2(T - T^*)}{T^*} \left[rT \left(1 - \frac{T}{K} \right) - nET \right] + \frac{2(E - E^*)}{E^*} [\sigma + \mu TE - \eta E].$$

Sustituyendo las ecuaciones del sistema, obtenemos

$$\frac{dV}{dt} = \frac{2(T - T^*)}{T^*} \left[rT \left(1 - \frac{T}{K} \right) - nET \right] + \frac{2(E - E^*)}{E^*} [\sigma + \mu TE - \eta E].$$

Simplificamos esta expresión para cada uno de los casos.

- $\mu > 0, n\sigma < r\eta$

Para $\mu > 0$, consideramos:

$$\frac{dV}{dt} = \frac{2(T - T^*)}{T^*} \left[rT \left(1 - \frac{T}{K} \right) - nET \right] + \frac{2(E - E^*)}{E^*} [\sigma + \mu TE - \eta E].$$

Dado que $T = T^*$ y $E = E^*$ son los puntos de equilibrio, analizamos las derivadas parciales de T y

E alrededor de estos puntos:

- La expresión $rT \left(1 - \frac{T}{K}\right)$ es negativa o cero para $T > 0$ y $T \approx T^*$.
- La expresión nET es positiva para $T > 0$ y $E > 0$.
- La expresión $\sigma + \mu TE - \eta E$ es negativa o cero para $E > 0$.

Esto implica que los términos negativos dominan, lo que lleva a $\frac{dV}{dt} \leq 0$. Por lo tanto, el punto de equilibrio P^* es globalmente asintóticamente estable en el primer caso.

- $\mu < 0, \eta + \mu K \geq 0, n\sigma < r\eta$

Para $\mu < 0$, consideramos:

$$\frac{dV}{dt} = \frac{2(T - T^*)}{T^*} \left[rT \left(1 - \frac{T}{K}\right) - nET \right] + \frac{2(E - E^*)}{E^*} [\sigma + \mu TE - \eta E].$$

Dado que $T = T^*$ y $E = E^*$ son los puntos de equilibrio, analizamos las derivadas parciales de T y E alrededor de estos puntos:

- La expresión $rT \left(1 - \frac{T}{K}\right)$ es negativa o cero para $T > 0$ y $T \approx T^*$.
- La expresión nET es positiva para $T > 0$ y $E > 0$.
- La expresión $\sigma + \mu TE - \eta E$ es negativa o cero para $E > 0$.

Esto implica que los términos negativos dominan, lo que lleva a $\frac{dV}{dt} \leq 0$. Por lo tanto, el punto de equilibrio P^* es globalmente asintóticamente estable en el segundo caso.

- $\mu < 0, \eta + \mu K < 0, n\sigma \leq r\eta$

Para $\mu < 0$, consideramos:

$$\frac{dV}{dt} = \frac{2(T - T^*)}{T^*} \left[rT \left(1 - \frac{T}{K}\right) - nET \right] + \frac{2(E - E^*)}{E^*} [\sigma + \mu TE - \eta E].$$

Dado que $T = T^*$ y $E = E^*$ son los puntos de equilibrio, analizamos las derivadas parciales de T y E alrededor de estos puntos:

- La expresión $rT \left(1 - \frac{T}{K}\right)$ es negativa o cero para $T > 0$ y $T \approx T^*$.
- La expresión nET es positiva para $T > 0$ y $E > 0$.

Caso	Condiciones	P_0	P^*	P_*	P_*^*
C1	$\mu > 0, n\sigma < r\eta$	IE	GAE-1	-	-
C2	$\mu < 0, \eta + \mu K \geq 0, n\sigma < r\eta$	IE	GAE-1	-	-
C3	$\mu < 0, \eta + \mu K < 0, n\sigma \leq r\eta$	IE	GAE-1	-	-
C4	$\mu < 0, \eta + \mu K < 0,$ $r\eta < n\sigma < -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$	LAE	LAE	IE	-
C5	$\mu < 0, \eta + \mu K < 0,$ $n\sigma = -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$	LAE	-	-	NS
C6	$\mu > 0, n\sigma \geq r\eta$	GAE-2	-	-	-
C7	$\mu < 0, \eta + \mu K < 0,$ $n\sigma > -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$	GAE-2	-	-	-

Tabla 2.1. La existencia y la estabilidad de todos los equilibrios factibles del modelo (2.2) sin retardo temporal. Aquí, IE significa inestable, GAE-1 globalmente asintóticamente estable en el primer cuadrante del plano T E , GAE-2 globalmente asintóticamente estable en el conjunto $\{(T, E) : T \geq 0, E > 0\}$, LAE localmente asintóticamente estable, NS nodo-silla.

- La expresión $\sigma + \mu TE - \eta E$ es negativa o cero para $E > 0$.

Esto implica que los términos negativos dominan, lo que lleva a $\frac{dV}{dt} \leq 0$. Por lo tanto, el punto de equilibrio P^* es globalmente asintóticamente estable en el tercer caso.

Para cerrar la Tabla (2.1) resume los resultados de esta sección, incluyendo cada caso donde probamos la existencia y no negatividad para los puntos de equilibrio y su estabilidad en dicho punto.

2.3. Análisis de estabilidad del modelo con retardo temporal.

Este análisis se basa principalmente en [30]. Para empezar recordemos nuevamente el modelo con retardo temporal.

$$\begin{cases} \frac{dT(t)}{dt} = rT(t) \left(1 - \frac{T(t)}{K}\right) - nE(t)T(t), \\ \frac{dE(t)}{dt} = \sigma + \mu T(t - \tau)E(t) - \eta E(t). \end{cases} \quad (2.6)$$

En esta sección, primero discutimos la estabilidad local del modelo (2.6), y luego investigamos la estabilidad global en una región factible.

2.3.1. Estabilidad local del modelo (2.6).

Es claro que los equilibrios factibles del modelo (2.6) son los mismos que los del modelo (2.2). Entonces la existencia de equilibrios se puede referir a la Tabla (2.1).

Para empezar, vamos a linealizar la ecuación (2.6) para cualquier equilibrio $\hat{P}(\hat{T}, \hat{E})$. En primer lugar, se

definen nuevas variables $x(t)$ y $y(t)$ que representan las pequeñas desviaciones de T y E respecto a sus valores de equilibrio \hat{T} y \hat{E} , respectivamente. Entonces, se reescriben las variables originales como $T(t) = \hat{T} + x(t)$ y $E(t) = \hat{E} + y(t)$ y se remplazan en la ecuación (2.6).

$$\begin{cases} \frac{d(\hat{T}+x(t))}{dt} = r(\hat{T}+x(t)) \left(1 - \frac{\hat{T}+x(t)}{K}\right) - n(\hat{E}+y(t))(\hat{T}+x(t)), \\ \frac{d(\hat{E}+y(t))}{dt} = \sigma + \mu(\hat{T}+x(t-\tau))(\hat{E}+y(t)) - \eta(\hat{E}+y(t)). \end{cases} \quad (2.19)$$

Simplificando (2.19) se obtiene el sistema

$$\begin{cases} \frac{dx(t)}{dt} = \left(r - \frac{2r\hat{T}}{K} - n\hat{E}\right)x(t) - n\hat{T}y(t), \\ \frac{dy(t)}{dt} = \mu\hat{E}x(t-\tau) - (\eta - \mu\hat{T})y(t). \end{cases} \quad (2.20)$$

Entonces, la ecuación característica de (2.20) en el equilibrio sin tumor $P_0\left(0, \frac{\sigma}{\eta}\right)$ las encontramos al remplazar $x(t) = c_1 e^{\lambda t}$ y $y(t) = c_2 e^{\lambda t}$ en el sistema linealizado, donde c_1 y c_2 son constantes.

$$\begin{cases} \lambda c_1 e^{\lambda t} = \left(r - \frac{2r\hat{T}}{K} - n\hat{E}\right)c_1 e^{\lambda t} - n\hat{T}c_2 e^{\lambda t}, \\ \lambda c_2 e^{\lambda t} = \mu\hat{E}c_1 e^{\lambda(t-\tau)} - (\eta - \mu\hat{T})c_2 e^{\lambda t}. \end{cases}$$

Remplazando P_0 y simplificando

$$\begin{cases} \lambda c_1 = \left(r - \frac{n\sigma}{\eta}\right)c_1, \\ \lambda c_2 = \mu\frac{\sigma}{\eta}c_1 e^{-\lambda\tau} - \eta c_2. \end{cases} \quad (2.21)$$

Podemos reorganizar las ecuaciones (2.21) para obtener una ecuación matricial

$$\begin{bmatrix} \lambda - \left(r - \frac{n\sigma}{\eta}\right) & 0 \\ \mu\frac{\sigma}{\eta}e^{-\lambda\tau} & \lambda + \eta \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Esta ecuación matricial representa un sistema homogéneo de ecuaciones lineales. Para que este sistema tenga soluciones no triviales, es decir, diferentes de cero, el determinante de la matriz de coeficientes debe ser cero. Esto nos lleva a la ecuación característica del sistema lineal

$$\left[\lambda - \left(r - \frac{n\sigma}{\eta}\right)\right](\lambda + \eta) = 0.$$

Por lo tanto, P_0 es localmente asintóticamente estable cuando $n\sigma > r\eta$, e inestable cuando $n\sigma < r\eta$. Esto es lo mismo que sin el retraso de tiempo en la Tabla (2.1).

Para un equilibrio tumoral $\hat{P}(\hat{T}, \hat{E})$ del modelo (2.6), \hat{T} y \hat{E} satisfacen el sistema

$$\begin{cases} r - \frac{rT}{K} - nE = 0, \\ \eta - \mu T = \frac{\sigma}{E}. \end{cases}$$

que se obtiene de (2.8). Entonces (2.20) puede reescribirse como

$$\begin{cases} \frac{dx}{dt} = -\frac{r\hat{T}}{K}x - n\hat{T}y, \\ \frac{dy}{dt} = \mu\hat{E}x(t - \tau) - \frac{\sigma}{\hat{E}}y. \end{cases} \quad (2.22)$$

Sustituyendo nuevamente $x = c_1 e^{\lambda t}$ e $y = c_2 e^{\lambda t}$ ($c_1^2 + c_2^2 \neq 0$) en el sistema lineal (2.22),

$$\begin{cases} \lambda c_1 = -\frac{r\hat{T}}{K}c_1 - n\hat{T}c_2, \\ \lambda c_2 = \mu\hat{E}c_1 e^{-\lambda\tau} - \frac{\sigma}{\hat{E}}c_2. \end{cases} \quad (2.23)$$

Podemos reorganizar las ecuaciones (2.23) para obtener una ecuación matricial

$$\begin{bmatrix} \lambda + \frac{r}{K}\hat{T} & -n\hat{T} \\ \mu\hat{E}e^{-\lambda\tau} & \lambda + \frac{\sigma}{\hat{E}} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

y con esto obtener la ecuación característica del sistema lineal (2.22) como

$$\left(\lambda + \frac{r}{K}\hat{T}\right)\left(\lambda + \frac{\sigma}{\hat{E}}\right) + n\mu\hat{E}\hat{T}e^{-\lambda\tau} = 0,$$

es decir,

$$\varphi(\lambda, \hat{T}, \hat{E}) := \lambda^2 + \left(\frac{r}{K}\hat{T} + \frac{\sigma}{\hat{E}}\right)\lambda + \hat{T}\left(\frac{r\sigma}{K\hat{E}} + n\mu\hat{E}e^{-\lambda\tau}\right) = 0. \quad (2.24)$$

Así, el equilibrio positivo \hat{P} es localmente asintóticamente estable si todas las raíces de (2.24) tienen partes reales negativas.

Sobre la base de las representaciones en (2.9), el cálculo directo produce

$$\varphi(0, T_*, E_*) = -rT_*\sqrt{\left(\mu - \frac{\eta}{K}\right)^2 + \frac{4n\sigma\mu}{Kr}} < 0.$$

Entonces, como $\lim_{\lambda \rightarrow \infty} \varphi(\lambda, T_*, E_*) = +\infty$, existe una raíz positiva de la ecuación $\varphi(\lambda, T_*, E_*) = 0$. Por lo tanto, el equilibrio más débil del tumor P_* es siempre inestable si existe. Esto es también lo mismo que en la Tabla (2.1).

Para P_* , la ecuación característica correspondiente es

$$\varphi(\lambda, T_*^*, E_*^*) = \lambda^2 + \left[\frac{r}{2\mu} \left(\mu + \frac{\eta}{K} \right) + \frac{2\sigma n\mu}{r} \left(\mu - \frac{\eta}{K} \right) \right] \lambda + \frac{Kr}{2} \left[\mu^2 - \left(\frac{\eta}{K} \right)^2 \right] (1 - e^{-\lambda\tau}).$$

Como $\lambda = 0$ es una raíz de $\varphi(\lambda, T_*^*, E_*^*) = 0$, P_* es un equilibrio de orden superior. Esto ocurre raramente en la realidad. Se omite la discusión posterior.

Para el equilibrio de tumor más fuerte P^* , la Tabla (2.1) muestra que es localmente asintóticamente estable cuando $\tau = 0$. Esto implica que todas las raíces de la ecuación (2.24) tienen partes reales negativas para $\tau = 0$.

En lo que sigue, solo consideramos la estabilidad de P^* para $\tau > 0$.

A medida que τ varía, la estabilidad del equilibrio del tumor P^* cambia solo cuando la ecuación trascendental (2.24) admite un par de raíces imaginarias puras. Entonces, sustituyendo $\lambda = i\omega$ con $\omega > 0$ en (2.24) y separando las partes real e imaginaria, obtenemos

$$\begin{cases} \omega^2 - \frac{r\sigma T^*}{KE^*} = n\mu E^* T^* \cos(\omega\tau), \\ \left(\frac{rT^*}{K} + \frac{\sigma}{E^*} \right) \omega = n\mu E^* T^* \sin(\omega\tau). \end{cases} \quad (2.25)$$

Elevando al cuadrado los dos lados de (2.25) y sumándolos, tenemos

$$F(\omega) := \omega^4 + \left[\left(\frac{rT^*}{K} \right)^2 + \left(\frac{\sigma}{E^*} \right)^2 \right] \omega^2 + T^{*2} \left(\frac{r\sigma}{KE^*} + n\mu E^* \right) \left(\frac{r\sigma}{KE^*} - n\mu E^* \right) = 0. \quad (2.26)$$

De (2.9) se deduce que

$$\frac{r\sigma}{KE^*} + n\mu E^* = \frac{r}{K} \sqrt{(\mu K - \eta)^2 + \frac{4n\sigma\mu K}{r}}$$

y

$$\frac{r\sigma}{KE^*} - n\mu E^* = -\frac{r(\mu K - \eta)}{K}.$$

Entonces $F(\omega)$ se puede reescribir como

$$F(\omega) = \omega^4 + \left[\left(\frac{rT^*}{K} \right)^2 + \left(\frac{\sigma}{E^*} \right)^2 \right] \omega^2 - \frac{r^2(\mu K - \eta)T^{*2}}{K^2} \sqrt{(\mu K - \eta)^2 + \frac{4n\sigma\mu K}{r}}. \quad (2.27)$$

Obviamente, $F(0) > 0$ para casos C2, C3 y C4 en la Tabla (2.1), ya que $\mu < 0$. Esto implica que $F(\omega) > 0$

para los casos C2, C3 y C4. Entonces, para estos casos, (2.24) no tiene raíces imaginarias puras al variar el retardo de tiempo; es decir, la estabilidad local de P^* es independiente del retardo de tiempo. Por lo tanto, para estos casos, P^* es absolutamente estable.

Cuando $0 < \mu K \leq \eta$, $F(\omega) > 0$ también se cumple para $\omega > 0$. Entonces, para el caso C1, P^* también es absolutamente estable cuando $0 < \mu K \leq \eta$.

Para el caso C1, cuando $\mu K > \eta$, $F(0) < 0$. Esto implica que $F(\omega) = 0$ tiene una única raíz positiva, denotada por ω_0 . Sustituyendo $\omega = \omega_0$ en (2.25), tenemos $\sin(\omega_0 \tau) > 0$. Además, a partir de la primera ecuación de (2.25), obtenemos

$$\tau = \tau_j = \frac{1}{\omega_0} \arccos \left[\frac{1}{n\mu E^* T^*} \left(\omega_0^2 - \frac{r\sigma T^*}{K E^*} \right) \right] + \frac{2j\pi}{\omega_0}, \quad j = 0, 1, 2, \dots \quad (2.28)$$

Por tanto, cuando $\tau = \tau_j$ ($j = 0, 1, 2, \dots$), la ecuación característica $\varphi(\lambda, T^*, E^*) = 0$ tiene un par de raíces imaginarias puras $\pm \omega_0 i$. Así, cuando τ aumenta y pasa por $\tau = \tau_j$, el cambio de estabilidad de P^* viene determinado por el signo de $\operatorname{Re} \left(\frac{d\lambda}{d\tau} \Big|_{\tau=\tau_j} \right)$.

De acuerdo con el Teorema 4.1 de [26], $\operatorname{Re} \left(\frac{d\lambda}{d\tau} \Big|_{\tau=\tau_j} \right)$ tiene el mismo signo que $F'(\omega_0)$. Podemos observar que $F'(\omega_0) > 0$. Entonces, la dirección de cruce es de izquierda a derecha a medida que τ aumenta y pasa por $\tau = \tau_j$. Obsérvese que $\tau = \tau_0$ es la primera vez que la raíz de $\varphi(\lambda, T^*, E^*) = 0$ cruza el eje imaginario al aumentar τ . Entonces P^* es localmente estable, asintóticamente cuando $0 < \tau < \tau_0$ e inestable cuando $\tau > \tau_0$. Correspondientemente, la bifurcación de Hopf (Sección A.4 Apéndice A) se produce en $\tau = \tau_0$ para el modelo (2.6), y las soluciones periódicas estables aparecen como $\tau > \tau_0$.

Resumiendo los resultados anteriores, tenemos los siguientes resultados con respecto a la estabilidad del modelo (2.6).

Teorema 2.2. *Para el modelo (2.6),*

1. *El equilibrio sin tumor P_0 es absolutamente estable si $n\sigma > r\eta$ y es inestable si $n\sigma < r\eta$.*
2. *El equilibrio con tumor más débil P_* (si existe) es siempre inestable.*
3. *El equilibrio del tumor P^* (si existe) es:*
 - *Absolutamente estable cuando $\mu < 0$ o $0 < \mu K \leq \eta$.*
 - *Localmente asintóticamente estable cuando $\mu K > \eta$ y $\tau < \tau_0$.*
 - *Inestable cuando $\mu K > \eta$ y $\tau > \tau_0$.*

2.3.2. Estabilidad global del modelo (2.6)

En esta sección investigamos la estabilidad global del equilibrio sin tumor P_0 y del equilibrio con tumor P^* . Dado que su estabilidad local se ha obtenido en la sección anterior, basta con demostrar que son respectivamente globalmente atractivos en la región factible bajo las condiciones correspondientes.

Para demostrar su atractivo, aplicamos diferentes métodos para P_0 y P^* . Para P_0 se utiliza el lema de fluctuación de [18]. Para P^* , su atractivo se obtiene construyendo una secuencia convergente mediante la técnica de iteración, que generaliza el método de [45]. En [45], se utiliza el Teorema de Squeeze que determina la convergencia de una secuencia. Aquí, además del Teorema de Squeeze, se aplica el Criterio de Acotamiento Monótono, puesto que la estructura del modelo (2.6) es diferente de la de [45].

Antes de probar la estabilidad global de P_0 , primero enunciamos el siguiente Lema de Fluctuación de [18] más específicamente el lema 4.2.

Lema 2.3. Sea $f : (t_0, \infty) \rightarrow \mathbb{R}$ acotada y continuamente diferenciable. Entonces hay secuencias

$$t_n, s_n \rightarrow \infty$$

con las siguientes propiedades:

$$f(t_n) \rightarrow f^\infty, \quad f'(t_n) \rightarrow 0;$$

$$f(s_n) \rightarrow f_\infty, \quad f'(s_n) \rightarrow 0$$

cuando $n \rightarrow \infty$, donde

$$f^\infty = \limsup_{t \rightarrow \infty} f(t) \quad \text{y} \quad f_\infty = \liminf_{t \rightarrow \infty} f(t).$$

Para la estabilidad global de P_0 , tenemos los siguientes resultados.

Teorema 2.4. Para el modelo (2.6), el equilibrio libre de tumores P_0 es globalmente estable asintóticamente en la región factible cuando se cumple una de las siguientes condiciones:

- (i) $\mu > 0$ y $n\sigma > r\eta$;
- (ii) $\mu < 0, \eta + \mu K \geq 0$ y $n\sigma \geq r\eta$;
- (iii) $\mu < 0, \eta + \mu K < 0$ y $n\sigma > -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$.

Demostración. Discutimos la estabilidad globalmente asintótica de P_0 para $\mu > 0$ y $\mu < 0$, respectivamente.

(i) Cuando $\mu > 0$, a partir de la segunda ecuación del modelo (2.6) tenemos $E' \geq \sigma - \eta E$ debido a la no negatividad de las soluciones. De ello se deduce que $\liminf_{t \rightarrow \infty} E(t) \geq \frac{\sigma}{\eta}$. Por tanto, cuando $n\sigma > r\eta$, para un número ε que satisfaga $0 < \varepsilon < \sigma - \frac{r\eta}{n}$, existe $t_0 > 0$ tal que $E(t) > \frac{\sigma - \varepsilon}{\eta}$ para $t > t_0$.

Además, para $t > t_0$, de la primera ecuación del modelo (2.6) se deduce que

$$\frac{dT}{dt} \leq T \left[r \left(1 - \frac{T}{K} \right) - \frac{n(\sigma - \varepsilon)}{\eta} \right] \leq \left[r - \frac{n(\sigma - \varepsilon)}{\eta} \right] T = \frac{n}{\eta} \left[\varepsilon - \left(\sigma - \frac{r\eta}{n} \right) \right] T.$$

Que $\varepsilon < \sigma - \frac{r\eta}{n}$ implica que $\lim_{t \rightarrow \infty} T(t) = 0$. De nuevo, aplicando $\lim_{t \rightarrow \infty} T = 0$ a la segunda ecuación del modelo (2.6) se obtiene $\lim_{t \rightarrow \infty} E(t) = \frac{\sigma}{\eta}$ por la teoría del sistema límite [41]. Por tanto, cuando $\mu > 0$ y $n\sigma > r\eta$, P_0 es globalmente atractiva. Por tanto, la estabilidad local de P_0 implica su estabilidad global si $\mu > 0$ y $n\sigma > r\eta$.

(ii) Para el caso $\mu < 0$, aplicamos el lema (2.3) para probar la atractividad de P_0 .

Cuando $\mu < 0$, de acuerdo con la no negatividad de las soluciones del modelo (2.6), tenemos

$$\frac{dT}{dt} \leq rT \left(1 - \frac{T}{K} \right), \text{ y } \frac{dE}{dt} \leq \sigma - \eta E.$$

Entonces $\limsup_{t \rightarrow \infty} T(t) \leq K$ y $\limsup_{t \rightarrow \infty} E(t) \leq \frac{\sigma}{\eta}$. Implica que el modelo (2.6) está acotado eventualmente para $\mu < 0$.

Por el Lema (2.3) se deduce que existen dos secuencias $\{t_n\}$ y $\{s_n\}$ tales que

$$\begin{aligned} \lim_{n \rightarrow \infty} t_n &= \infty, \quad \lim_{n \rightarrow \infty} T(t_n) = T^\infty, \quad \lim_{n \rightarrow \infty} \frac{dT(t_n)}{dt} = 0, \\ \lim_{n \rightarrow \infty} s_n &= \infty, \quad \lim_{n \rightarrow \infty} E(s_n) = E^\infty, \quad \lim_{n \rightarrow \infty} \frac{dE(s_n)}{dt} = 0. \end{aligned} \tag{2.29}$$

De (2.6), tenemos

$$\begin{aligned} \frac{dT(t_n)}{dt} &= T(t_n) \left[r \left(1 - \frac{T(t_n)}{K} \right) - nE(t_n) \right], \\ \frac{dE(s_n)}{dt} &= \sigma + \mu T(s_n - \tau) E(s_n) - \eta E(s_n). \end{aligned} \tag{2.30}$$

Entonces, dejando que $n \rightarrow \infty$, a partir de (2.29) y (2.30) tenemos

$$\begin{aligned}
0 &\leq T^\infty \left[r \left(1 - \frac{T^\infty}{K} \right) - nE_\infty \right], \\
0 &\geq \sigma + \mu T^\infty E_\infty - \eta E_\infty.
\end{aligned} \tag{2.31}$$

Afirmamos que $T^\infty = 0$. Si no, $T^\infty > 0$ ya que $0 \leq T^\infty \leq K$. De la primera desigualdad de (2.31) se deduce que

$$nE_\infty \leq r \left(1 - \frac{T^\infty}{K} \right).$$

Además, de la segunda desigualdad de (2.31) se deduce que

$$n\sigma + r(\mu T^\infty - \eta) \left(1 - \frac{T^\infty}{K} \right) \leq 0. \tag{2.32}$$

Obsérvese que, cuando se cumple cualquiera de las condiciones (ii) y (iii) del Teorema (2.4), la función cuadrática de T , $n\sigma + r(\mu T - \eta) \left(1 - \frac{T}{K} \right)$, es positiva en el intervalo $(0, K]$. Entonces la desigualdad (2.32) no se cumple bajo ninguna de las condiciones (ii) y (iii). Por lo tanto, la afirmación es cierta. Implica que $\lim_{t \rightarrow \infty} T(t) = 0$.

Además, aplicando $\lim_{t \rightarrow \infty} T(t) = 0$ a la segunda ecuación del modelo (2.6), podemos obtener fácilmente $\lim_{t \rightarrow \infty} E(t) = \frac{\sigma}{\eta}$ por la teoría del sistema límite [41]. Por tanto, para los casos (ii) y (iii), la estabilidad local de P_0 implica también su estabilidad global.

La demostración del Teorema (2.4) está completa.

Con respecto a la estabilidad global de P^* del modelo (2.6), tenemos los siguientes resultados.

Teorema 2.5. *El equilibrio tumoral P^* del modelo (2.6) es global asintóticamente estable si se cumple alguna de las dos condiciones siguientes:*

- (i) $\eta > \mu K > 0$ y $n\sigma < r\eta$;
- (ii) $\mu < 0$ y $n\sigma < r\eta$.

Para demostrar la atractividad global del equilibrio tumoral P^* del modelo (2.6) bajo las condiciones del Teorema (2.5), necesitamos construir dos secuencias convergentes diferentes correspondientes a los dos casos, $\mu > 0$ y $\mu < 0$, mediante la técnica de iteración. A continuación, definimos las dos secuencias convergentes con el mismo método general pero diferentes términos, y describimos sus propiedades con dos lemas.

Lema 2.6. Supongamos $n\sigma < r\eta$. Entonces hay dos sucesiones $\{\tilde{T}_m\}$ y $\{\tilde{E}_m\}$ ($m = 0, 1, 2, \dots$), definidas por:

$$\tilde{E}_0 = \frac{\sigma}{\eta}, \quad \tilde{T}_m = K \left(1 - \frac{n\tilde{E}_m}{r} \right), \quad \tilde{E}_{m+1} = \frac{\sigma}{\eta - \mu\tilde{T}_m} \quad (2.33)$$

tienen las siguientes propiedades

(i) Cuando $0 < \mu K < \eta$,

$$0 < \tilde{T}_m < K \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu} \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu}, \quad \frac{\sigma}{\eta} < \tilde{E}_m < \frac{r}{n}, \quad m = 1, 2, \dots$$

(ii) Cuando $\mu < 0$,

$$\tilde{T}_m > 0, \quad 0 < \tilde{E}_m < \frac{\sigma}{\eta} < \frac{r}{n}, \quad m = 1, 2, \dots$$

(iii) Para $0 < \mu K < \eta$ or $\mu < 0$,

$$\lim_{m \rightarrow \infty} \tilde{T}_m = T^*, \quad \lim_{m \rightarrow \infty} \tilde{E}_m = E^*$$

Donde T^* y E^* corresponden a las coordenadas de P^* .

Demostración. (i) Primero, cuando $n\sigma < r\eta$ y $0 < \mu K < \eta$, tenemos

$$0 < \tilde{T}_0 = K \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu} \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu}.$$

Por (2.33), $0 < \tilde{T}_0 < \frac{\eta}{\mu}$ implica que $\tilde{E}_1 > \frac{\sigma}{\eta}$, y $\tilde{T}_0 < \frac{\eta}{\mu} \left(1 - \frac{n\sigma}{r\eta} \right)$ implica $\tilde{E}_1 < \frac{r}{n}$. Es decir, $\frac{\sigma}{\eta} < \tilde{E}_1 < \frac{r}{n}$. De ello se deduce que $0 < \tilde{T}_1 < K \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu} \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu}$.

Por inducción adicional, sabemos que $0 < \tilde{T}_m < K \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu} \left(1 - \frac{n\sigma}{r\eta} \right) < \frac{\eta}{\mu}$ y $\frac{\sigma}{\eta} < \tilde{E}_m < \frac{r}{n}$ ($m = 1, 2, 3, \dots$), cuando $n\sigma < r\eta$ y $0 < \mu K < \eta$.

A continuación, demostramos la existencia de los límites de las secuencias $\{\tilde{T}_m\}$ y $\{\tilde{E}_m\}$. Un cálculo sencillo muestra

$$\tilde{E}_{m+2} = \frac{\frac{\sigma}{\eta} \left[\frac{\mu n K}{r\eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta} \right) \right]}{\left(1 - \frac{\mu K}{\eta} \right) \frac{\mu n K}{r\eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta} \right)^2 + \frac{\mu n \sigma K}{r\eta^2}} \quad (2.34)$$

y

$$\tilde{E}_{m+2} - E^* = \frac{\left[\frac{\sigma}{\eta} - \left(1 - \frac{\mu K}{\eta}\right) E^* \right] \left(\frac{\mu n K}{r \eta} \tilde{E}_m + 1 - \frac{\mu K}{\eta} \right) - \frac{\sigma}{\eta} \cdot \frac{\mu n K}{r \eta} E^*}{\left(1 - \frac{\mu K}{\eta}\right) \frac{\mu n K}{r \eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta}\right)^2 + \frac{\mu n \sigma K}{r \eta^2}}.$$

Según (2.8), E^* es la raíz positiva de la ecuación

$$h(E) = \frac{\mu n K}{r \eta} E^2 + \left(1 - \frac{\mu K}{\eta}\right) E - \frac{\sigma}{\eta} = 0,$$

es decir,

$$\frac{\sigma}{\eta} = \frac{\mu n K}{r \eta} E^{*2} + \left(1 - \frac{\mu K}{\eta}\right) E^*.$$

Entonces

$$\tilde{E}_{m+2} - E^* = \frac{(\mu n K)^2 E^{*2} (\tilde{E}_m - E^*)}{(r \eta)^2 \left[\left(1 - \frac{\mu K}{\eta}\right) \frac{\mu n K}{r \eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta}\right)^2 + \frac{\mu n \sigma K}{r \eta^2} \right]}. \quad (2.35)$$

Note que

$$h(\tilde{E}_0) = \frac{\mu K}{\eta} \tilde{E}_0 \left(\frac{n}{r} \tilde{E}_0 - 1 \right) = \frac{\mu \sigma K}{\eta^2} \left(\frac{n \sigma}{r \eta} - 1 \right),$$

y

$$\begin{aligned} h(\tilde{E}_1) &= \frac{\mu K}{\eta} \tilde{E}_1 \left(\frac{n}{r} \tilde{E}_1 - 1 \right) + \left(\tilde{E}_1 - \frac{\sigma}{\eta} \right), \\ &= \frac{\mu K}{\eta} \cdot \frac{\sigma}{\eta - \mu \tilde{T}_0} \left(\frac{n}{r} \cdot \frac{\sigma}{\eta - \mu \tilde{T}_0} - 1 \right) + \left(\frac{\sigma}{\eta - \mu \tilde{T}_0} - \frac{\sigma}{\eta} \right), \\ &= \frac{n \mu^2 \sigma^2 K \tilde{T}_0}{r \eta^2 (\eta - \mu \tilde{T}_0)^2} > 0. \end{aligned}$$

Entonces $\tilde{E}_0 < E^* < \tilde{E}_1$ según la propiedad de la función $h(E)$. Además, la recursión de (2.35) muestra que $0 < \tilde{E}_{2k} < E^* < \tilde{E}_{2k-1} (k = 1, 2, \dots)$. Por otro lado,

$$\begin{aligned} \tilde{E}_{m+2} - \tilde{E}_m &= - \frac{\left(1 - \frac{\mu K}{\eta}\right) \left[\frac{\mu n K}{r \eta} \tilde{E}_m^2 + \left(1 - \frac{\mu K}{\eta}\right) \tilde{E}_m - \frac{\sigma}{\eta} \right]}{\left(1 - \frac{\mu K}{\eta}\right) \frac{\mu n K}{r \eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta}\right)^2 + \frac{\mu n \sigma K}{r \eta^2}}, \\ &= - \frac{\left(1 - \frac{\mu K}{\eta}\right) h(\tilde{E}_m)}{\left(1 - \frac{\mu K}{\eta}\right) \frac{\mu n K}{r \eta} \tilde{E}_m + \left(1 - \frac{\mu K}{\eta}\right)^2 + \frac{\mu n \sigma K}{r \eta^2}}. \end{aligned}$$

Entonces, $0 < \tilde{E}_{2k} < E^* < \tilde{E}_{2k-1}$ implica $h(\tilde{E}_{2k}) < 0$ y $h(\tilde{E}_{2k-1}) > 0 (k = 1, 2, \dots)$. Por lo tanto, la secuencia $\{\tilde{E}_{2k}\}$ es monótona creciente, y la secuencia $\{\tilde{E}_{2k-1}\}$ es monotoma decreciente. Por eso, cuando $n \sigma < r \eta$ y $0 < \mu K < \eta$, los límites de las secuencias $\{\tilde{E}_{2k}\}$ y $\{\tilde{E}_{2k-1}\}$ existen.

Por último, dejar que $m \rightarrow \infty$ para los dos lados de (2.34) consigue que los límites de $\{\tilde{E}_{2k}\}$ y $\{\tilde{E}_{2k-1}\}$

ambas satisfacen $h(E) = 0$. Por lo tanto sabemos que $\lim_{k \rightarrow \infty} \tilde{E}_{2k} = \lim_{k \rightarrow \infty} \tilde{E}_{2k-1} = E^*$ ya que E^* es la única raíz positiva de la ecuación $h(E) = 0$.

Además, a partir de $\tilde{T}_m = K \left(1 - \frac{n\tilde{E}_m}{r}\right)$ se deduce que $\lim_{m \rightarrow \infty} \tilde{T}_m = T^*$.

(ii) De $n\sigma < r\eta$ y $\mu < 0$ se sigue que

$$\tilde{T}_0 = K \left(1 - \frac{n\tilde{E}_0}{r}\right) = K \left(1 - \frac{n\sigma}{r\eta}\right) > 0$$

y

$$\tilde{E}_1 = \frac{\sigma}{\eta - \mu\tilde{T}_0} < \frac{\sigma}{\eta} < \frac{r}{n}.$$

De nuevo, $\tilde{E}_1 < \frac{r}{n}$ implica $\tilde{T}_1 > 0$. Por lo tanto, por inducción sabemos $\tilde{E}_m < \frac{\sigma}{\eta} < \frac{r}{n}$ y $\tilde{T}_m > 0$ ($m = 1, 2, \dots$) as $n\sigma < r\eta$ y $\mu < 0$. Por otra parte, $\tilde{T}_m > 0$ implica $\tilde{E}_{m+1} > 0$.

Un cálculo sencillo muestra

$$\tilde{E}_{m+2} - \tilde{E}_{m+1} = \frac{\sigma\mu(\tilde{T}_{m+1} - \tilde{T}_m)}{(\eta - \mu\tilde{T}_{m+1})(\eta - \mu\tilde{T}_m)} = \frac{\sigma\mu nK(\tilde{E}_m - \tilde{E}_{m+1})}{r(\eta - \mu\tilde{T}_{m+1})(\eta - \mu\tilde{T}_m)}.$$

Entonces $\tilde{E}_1 < \tilde{E}_0$ implica $\tilde{E}_2 < \tilde{E}_1$. Por recursión, $\tilde{T}_m > 0$ supone que la secuencia $\{\tilde{E}_m\}$ es monotona decreciente. Por tanto, el límite de \tilde{E}_m existe. Análogamente, para el caso $n\sigma < r\eta$ y $0 < \mu K < \eta$, tenemos $\lim_{m \rightarrow \infty} \tilde{T}_m = T^*$ y $\lim_{m \rightarrow \infty} \tilde{E}_m = E^*$.

La prueba del Lema (2.6) está completa.

Lema 2.7. Cuando $n\sigma < r\eta$ y $\mu < 0$, las dos sucesiones $\{\tilde{T}'_m\}$ y $\{\tilde{E}'_m\}$ ($m = 0, 1, 2, \dots$), definidas por

$$\tilde{T}'_0 = K, \quad \tilde{E}'_m = \frac{\sigma}{\eta - \mu\tilde{T}'_m}, \quad \tilde{T}'_{m+1} = K \left(1 - \frac{n\tilde{E}'_m}{r}\right) \quad (2.36)$$

tienen las siguientes propiedades:

$$\tilde{T}'_m > 0, \quad 0 < \tilde{E}'_m < \frac{\sigma}{\eta} < \frac{r}{n}$$

y

$$\lim_{m \rightarrow \infty} \tilde{T}'_m = T^*, \quad \lim_{m \rightarrow \infty} \tilde{E}'_m = E^*.$$

Cuando T^* y E^* corresponden a las coordenadas de P^* .

Demostración. Observe que las relaciones recursivas entre las dos secuencias definidas en los lemas (2.6) y (2.7) son las mismas, pero sus términos iniciales son diferentes.

Para el lema (2.7), $0 < \tilde{E}'_0 = \frac{\sigma}{\eta - \mu K} < \frac{\sigma}{\eta} < \frac{r}{n}$. Correspondientemente, $\tilde{T}'_1 > 0$. Entonces por inducción para las dos secuencias podemos demostrar fácilmente que $\tilde{T}'_m > 0$ y $0 < \tilde{E}'_m < \frac{\sigma}{\eta} < \frac{r}{n}$ ($m = 0, 1, 2, \dots$) cuando $n\sigma < r\eta$ y $\mu < 0$.

Note que

$$\tilde{E}'_1 - \tilde{E}'_0 = \frac{\sigma\mu(\tilde{T}'_1 - \tilde{T}'_0)}{(\eta - \mu\tilde{T}'_1)(\eta - \mu\tilde{T}'_0)} = -\frac{\sigma\mu nK\tilde{E}'_0}{r(\eta - \mu\tilde{T}'_1)(\eta - \mu\tilde{T}'_0)} > 0.$$

Entonces, de

$$\tilde{E}'_{m+2} - \tilde{E}'_{m+1} = \frac{\sigma\mu(\tilde{T}'_{m+2} - \tilde{T}'_{m+1})}{(\eta - \mu\tilde{T}'_{m+2})(\eta - \mu\tilde{T}'_{m+1})} = \frac{\sigma\mu nK(\tilde{E}'_m - \tilde{E}'_{m+1})}{r(\eta - \mu\tilde{T}'_{m+2})(\eta - \mu\tilde{T}'_{m+1})}.$$

Sabemos que la secuencia $\{\tilde{E}'_m\}$ es monótona creciente por recursión. Por lo tanto, de forma similar a la demostración del Lema (2.6), $\lim_{m \rightarrow \infty} \tilde{T}'_m = T^*$ y $\lim_{m \rightarrow \infty} \tilde{E}'_m = E^*$.

Esto completa prueba del Lema (2.7).

Con los lemas (2.6) y (2.6) ya tenemos las herramientas para demostrar el teorema (2.5).

Demostración. (i) Cuando $\mu > 0$, a partir de la segunda ecuación del modelo (2.6) tenemos $E' \geq \sigma - \eta E$ debido a la no negatividad de las soluciones del modelo (2.6). Se deduce que $\liminf_{t \rightarrow \infty} E(t) \geq \frac{\sigma}{\eta}$. Entonces, para $\varepsilon > 0$, suficientemente pequeño, existe $t_0 > 0$ tal que $E(t) > \frac{\sigma}{\eta} - \varepsilon := E_0(\varepsilon)$ para $t > t_0$.

A continuación, de la primera ecuación del modelo (2.6) se deduce que, para $t > t_0$,

$$T' \leq T \left[r \left(1 - \frac{T}{K} \right) - nE_0(\varepsilon) \right] = \frac{rT}{K} [\hat{T}_0(\varepsilon) - T],$$

donde $\hat{T}_0(\varepsilon) = K \left[1 - \frac{nE_0(\varepsilon)}{r} \right]$. Obsérvese que $\lim_{\varepsilon \rightarrow 0} \hat{T}_0(\varepsilon) = \tilde{T}_0 \in (0, \frac{\eta}{\mu})$ para $n\sigma < r\eta$ y $\eta > \mu K > 0$, donde \tilde{T}_0 se define en (2.33). Entonces hay ε , suficientemente pequeño, tal que $\hat{T}_0(\varepsilon) \in (0, \frac{\eta}{\mu})$. De ello se desprende que $\limsup_{t \rightarrow \infty} T(t) \leq \hat{T}_0(\varepsilon)$. Por lo tanto, hay t'_0 ($t'_0 > t_0$) tal que $T(t) < \hat{T}_0(\varepsilon) + \varepsilon := T_0(\varepsilon)$ para $t > t'_0$.

De nuevo, para $t > t'_0 + \tau$, a partir de la segunda ecuación de (2.6) tenemos

$$E' \leq \sigma - [\eta - \mu T_0(\varepsilon)] E = [\eta - \mu T_0(\varepsilon)] [\hat{E}_1(\varepsilon) - E],$$

donde $\hat{E}_1(\varepsilon) = \frac{\sigma}{\eta - \mu T_0(\varepsilon)}$. Aquí, $\lim_{\varepsilon \rightarrow 0} \hat{E}_1(\varepsilon) = \tilde{E}_1 \in (\frac{\sigma}{\eta}, \frac{r}{n})$, donde \tilde{E}_1 es definido en (2.33). De manera similar, hay un t_1 ($t_1 > t'_0 + \tau$) tal que $E(t) < \hat{E}_1(\varepsilon) + \varepsilon := E_1(\varepsilon)$ para $t > t_1$.

Además, para $t > t_1$, obtenemos de la primera ecuación de (2.6)

$$T' \geq T \left[r \left(1 - \frac{T}{K} \right) - nE_1(\varepsilon) \right] = \frac{rT}{K} [\hat{T}_1(\varepsilon) - T],$$

donde $\hat{T}_1(\varepsilon) = K \left[1 - \frac{nE_1(\varepsilon)}{r} \right]$. De la misma forma, hay un t'_1 ($t'_1 > t_1$) tal que $T(t) > \hat{T}_1(\varepsilon) - \varepsilon := T_1(\varepsilon)$ para $t > t'_1$.

Hasta aquí obtenemos $E_0(\varepsilon), T_0(\varepsilon), E_1(\varepsilon)$ y $T_1(\varepsilon)$. Repitiendo el proceso anterior podemos obtener las dos secuencias $\{E_m(\varepsilon)\}$ y $\{T_m(\varepsilon)\}$ ($m = 0, 1, 2, \dots$). Según el proceso de obtención de las dos secuencias, para ε dado, suficientemente pequeño, y el correspondiente t^* suficientemente grande, tenemos, para $t > t^*$,

$$\begin{aligned} cE_{2m}(\varepsilon) &< E(t) < E_{2m+1}(\varepsilon), \\ T_{2m+1}(\varepsilon) &< T(t) < E_{2m}(\varepsilon). \end{aligned} \quad m = 0, 1, 2, \dots \quad (2.37)$$

Observe que ε es arbitrariamente suficientemente pequeño, y que $\lim_{\varepsilon \rightarrow 0} E_m(\varepsilon) = \tilde{E}_m$ y $\lim_{\varepsilon \rightarrow 0} T_m(\varepsilon) = \tilde{T}_m$, donde \tilde{E}_m y \tilde{T}_m están definidas en (2.33). Entonces, de (2.37) se deduce que

$$\begin{aligned} \tilde{E}_{2m} &\leq \liminf_{t \rightarrow \infty} E(t) \leq \limsup_{t \rightarrow \infty} E(t) \leq \tilde{E}_{2m+1}, \\ \tilde{T}_{2m+1} &\leq \liminf_{t \rightarrow \infty} T(t) \leq \limsup_{t \rightarrow \infty} T(t) \leq \tilde{T}_{2m}, \quad m = 0, 1, 2, \dots \end{aligned} \quad (2.38)$$

El lema (2.6) muestra que $\lim_{m \rightarrow \infty} E_m = E^*$ y $\lim_{m \rightarrow \infty} \tilde{T}_m = T^*$. Entonces resulta que $\lim_{t \rightarrow \infty} E(t) = E^*$ y $\lim_{t \rightarrow \infty} T(t) = T^*$ según (2.38). Es decir, P^* es globalmente atractivo.

Por lo tanto, la estabilidad local de P^* implica que P^* es globalmente estable asintóticamente cuando $n\sigma < r\eta$ y $\eta > \mu K > 0$.

(ii) Cuando $\mu < 0$, por la derivación similar al caso anterior que $\mu > 0$, para las mismas dos secuencias $\{E_m(\varepsilon)\}$ y $\{T_m(\varepsilon)\}$ ($m = 0, 1, 2, \dots$), sabemos que existe un t'_m tal que $E(t) < E_m(\varepsilon)$ y $T(t) > T_m(\varepsilon)$ para $t > t'_m$. Así, puesto que ε es arbitrariamente suficientemente pequeño, tenemos $E(t) \leq \tilde{E}_m$ y $T(t) \geq \tilde{T}_m$. Además, se deduce que $\limsup_{t \rightarrow \infty} E(t) \leq E^*$ y $\liminf_{t \rightarrow \infty} T(t) \geq T^*$ ya que $\lim_{\mu \rightarrow \infty} \tilde{E}_m = E^*$ y $\lim_{\mu \rightarrow \infty} \tilde{T}_m = T^*$ por el Lema (2.7).

Por otro lado, para obtener $\liminf_{t \rightarrow \infty} E(t) \geq E^*$ y $\limsup_{t \rightarrow \infty} T(t) \leq T^*$, construimos otras dos secuencias $\{E'_m(\varepsilon)\}$ y $\{T'_m(\varepsilon)\}$ ($m = 0, 1, 2, \dots$) como sigue.

De la primera ecuación del modelo (2.6) tenemos $T' \leq rT \left(1 - \frac{T}{K} \right)$. Se deduce que $\limsup_{t \rightarrow \infty} T(t) \leq$

K . Entonces, para $\varepsilon > 0$, suficientemente pequeño, existe un $\bar{t}_0 > 0$ tal que $T(t) < K + \varepsilon := T'_0(\varepsilon)$ para $t > \bar{t}_0$.

Para $\mu < 0$, de la segunda ecuación del modelo (2.6) tenemos, para $t > \bar{t}_0 + \tau$,

$$E' \geq \sigma - [\eta - \mu T'_0(\varepsilon)] E = [\eta - \mu T'_0(\varepsilon)] [\hat{E}'_0(\varepsilon) - E].$$

Donde $\hat{E}'_0(\varepsilon) = \frac{\sigma}{\eta - \mu T'_0(\varepsilon)}$. Nótese que $\lim_{\varepsilon \rightarrow 0} \hat{E}'_0(\varepsilon) = \tilde{E}'_0$ con $0 < E'_0 < \frac{\sigma}{\eta} < \frac{r}{n}$, donde \tilde{E}'_0 se define en (2.36). Entonces hay ε , suficientemente pequeño, tal que $\hat{E}'_0(\varepsilon) < \frac{\sigma}{\eta} < \frac{r}{n}$. De ello se deduce que $\liminf_{t \rightarrow \infty} E(t) \geq \hat{E}'_0(\varepsilon)$. Por lo tanto, hay \bar{t}'_0 ($\bar{t}'_0 > \bar{t}_0$) tal que $E(t) > \hat{E}'_0(\varepsilon) - \varepsilon := E'_0(\varepsilon)$ para $t > \bar{t}'_0$.

Además, para $t > \bar{t}'_0$, obtenemos de la primera ecuación de (2.6)

$$T' \leq T \left[r \left(1 - \frac{T}{K} \right) - n E'_0(\varepsilon) \right] = \frac{rT}{K} [\hat{T}'_1(\varepsilon) - T],$$

donde $\hat{T}'_1(\varepsilon) = K \left[1 - \frac{n E'_0(\varepsilon)}{r} \right]$. Del mismo modo, hay \bar{t}_1 ($\bar{t}_1 > \bar{t}'_0$) tal que $T(t) < \hat{T}'_1(\varepsilon) + \varepsilon := T'_1(\varepsilon)$ para $t > \bar{t}_1$.

De nuevo, para $t > \bar{t}_1 + \tau$, a partir de la segunda ecuación de (2.6) tenemos

$$E' \geq \sigma - [\eta - \mu T'_1(\varepsilon)] E = [\eta - \mu T'_1(\varepsilon)] [\hat{E}'_1(\varepsilon) - E],$$

donde $\hat{E}'_1(\varepsilon) = \frac{\sigma}{\eta - \mu T'_1(\varepsilon)}$. Repitiendo el proceso anterior, hay un \bar{t}'_1 ($\bar{t}'_1 > \bar{t}_1$) tal que $E(t) > \hat{E}'_1(\varepsilon) - \varepsilon := E'_1(\varepsilon)$ para $t > \bar{t}'_1$. Así, obtenemos $T'_0(\varepsilon)$, $E'_0(\varepsilon)$, $T'_1(\varepsilon)$ y $E'_1(\varepsilon)$. Continuando de esta manera, podemos obtener las dos secuencias $\{T'_m(\varepsilon)\}$ y $\{E'_m(\varepsilon)\}$ ($m = 0, 1, 2, \dots$), y existe \tilde{t}_m tal que $E(t) > E_m(\varepsilon)$ y $T(t) < T_m(\varepsilon)$ para $t > \tilde{t}_m$. Implica que $\liminf_{t \rightarrow \infty} E(t) \geq E_m(\varepsilon)$ y $\limsup_{t \rightarrow \infty} T(t) \leq T_m(\varepsilon)$. Dado que ε es arbitrariamente suficientemente pequeño, tenemos que $\liminf_{t \rightarrow \infty} E(t) \geq \tilde{E}'_m$ y $\limsup_{t \rightarrow \infty} T(t) \leq \tilde{T}'_m$, donde \tilde{E}'_m y \tilde{T}'_m se definen en (2.36.)

Nótese que el Lema (2.7) muestra que $\lim_{m \rightarrow \infty} \tilde{E}'_m = E^*$ y $\lim_{m \rightarrow \infty} \tilde{T}'_m = T^*$. Entonces,

$$\liminf_{t \rightarrow \infty} E(t) \geq E^* \quad \text{y} \quad \limsup_{t \rightarrow \infty} T(t) \leq T^*.$$

Construyendo las dos secuencias diferentes, las inferencias anteriores han demostrado que

$$\limsup_{t \rightarrow \infty} E(t) \leq E^* \quad \text{y} \quad \liminf_{t \rightarrow \infty} T(t) \geq T^*$$

y que $\liminf_{t \rightarrow \infty} E(t) \geq E^*$ y $\limsup_{t \rightarrow \infty} T(t) \leq T^*$. Por tanto, $\lim_{t \rightarrow \infty} E(t) = E^*$ y $\lim_{t \rightarrow \infty} T(t) =$

T^* .

Esto completa la demostración del Teorema (2.5).

Capítulo 3

Simulaciones numéricas

En esta sección, ilustramos el efecto del retardo temporal del modelo (2.6) cuando existe un equilibrio tumoral, y mostramos el comportamiento de (2.6) mediante simulaciones numéricas que hemos realizado con un módulo en python llamado `jitcdde` diseñado para resolver ecuaciones diferenciales con retraso el enlace al código con todas las implementaciones se encuentra en el apéndice C. La estructura del modelo (2.6) muestra que la existencia de sus equilibrios es independiente del retardo de tiempo, y los resultados obtenidos en la sección anterior ilustran que la demora de tiempo puede afectar a la estabilidad del modelo (2.6) solo cuando $\mu K > \eta$ y $n\sigma < r\eta$ tal que el tumor de equilibrio P^* existe. Es decir, bajo la condición de que $\mu K > \eta$ y $n\sigma < r\eta$, existe un τ_0 (definido en (2.28)) tal que P^* es estable cuando $\tau < \tau_0$ e inestable cuando $\tau > \tau_0$. Aquí, la bifurcación de Hopf se produce para el modelo (2.6) cuando τ aumenta y cruza τ_0 .

Adicionalmente, aparece una solución periódica estable para $\tau > \tau_0$ (Figura (3.1)), y el diagrama de bifurcación del modelo (2.6) se muestra en la Figura (3.3) y en (3.2) podemos ver como cambian los valores para diferentes valores de τ incluso una versión sin retardo con $\tau = 0$. En las figuras (3.1), (3.2) y (3.3), los valores de los parámetros del modelo (2.6) son $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 4$ y $\eta = 1,5$. Estos valores de los parámetros satisfacen las condiciones $\mu K > \eta$ y $n\sigma < r\eta$. En este caso, el equilibrio tumoral es $P^*(0,3272, 2,6138)$, y el umbral que determina la estabilidad de P^* es $\tau_0 = 0,2242$ de (2.28). En la figura (3.1), $\tau = 0,3$. La figura (3.3) muestra que, cuando $\tau > \tau_0$, la amplitud de la solución periódica aumenta con el tiempo de retardo.

Cuando $0 < \mu < \frac{\eta}{K}$, el equilibrio tumoral P^* es absolutamente estable por el Teorema (2.4) si existe (es decir, $n\sigma < r\eta$). Tomando tres valores diferentes del retardo, $\tau = 1, 5, 10$, respectivamente, la Figura (3.4) muestra que el retardo ralentiza la convergencia de las soluciones a P^* del modelo (2.6). Aquí, $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 0,4$ y $\eta = 1,5$, y el modelo (2.6) tiene un único equilibrio tumoral

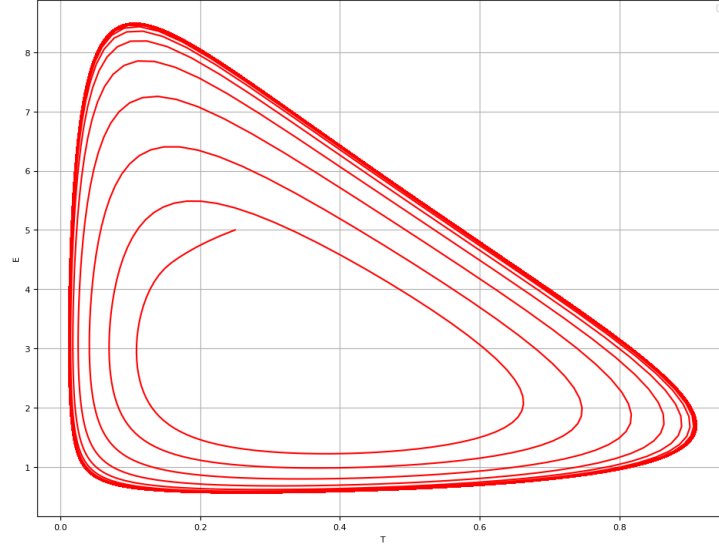


Figura 3.1. La órbita periódica estable del modelo (2.6) a medida que $\mu K > \eta$ y $n\sigma < r\eta$. Aquí, $r = 2,5$, $K = 2$, $n = 0,8$, $\sigma = 0,5$, $\mu = 4$, $\eta = 1,5$ y $\tau = 0,3 > 0,2242 = \tau_0$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0,3272, 2,6138)$, y una órbita periódica estable.

$P^*(1,6238, 0,5879)$ que es globalmente estable asintóticamente.

En las siguientes simulaciones, exploramos otros casos del modelo (2.6) bajo diferentes condiciones de parámetros y retardos temporales. Primero, analizamos el caso donde $\mu < 0$, $\eta + \mu K \geq 0$, y $n\sigma < r\eta$. En este escenario, la Figura (3.5) muestra las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$. Aquí, los valores de los parámetros son $r = 2,5$, $K = 0,5$, $n = 0,8$, $\sigma = 0,5$, $\mu = -0,5$, $\eta = 1,5$. Adicionalmente, consideramos el caso donde $\mu < 0$, $\eta + \mu K < 0$, y $n\sigma \leq r\eta$. En este escenario, la Figura (3.6) muestra las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$. Aquí, los valores de los parámetros son $r = 2,5$, $K = 3,0$, $n = 0,8$, $\sigma = 0,5$, $\mu = -2,5$, $\eta = 1,5$. Para el caso $\mu < 0$, el Teorema (2.4) muestra que el retardo temporal no afecta a la estabilidad de los equilibrios del modelo (2.6), pero, al igual que en el caso $0 < \mu < \frac{\eta}{K}$, podría desempeñar un papel que afecte a la convergencia de las soluciones de (2.6). Aquí, mostramos los efectos del retraso en las soluciones del modelo (2.6) solo para el caso C4 en la Tabla (2.1); es decir, hay dos equilibrios tumorales P^* y P_* del modelo (2.6) (Figura (3.7)). Para la figura (3.7), $r = 2,5$, $K = 4$, $n = 0,8$, $\sigma = 6,2$, $\mu = -1,2$ y $\eta = 1,5$, que satisfacen la condición C4 de la tabla (2.1). En correspondencia con el conjunto de valores de los parámetros, el modelo (2.6) tiene dos equilibrios positivos $P^*(1,9016, 1,6394)$ y $P_*(0,8484, 2,4622)$ con P^* estable y P_* inestable. En este caso, donde las células tumorales neutralizan las células efectoras y el modelo (2.6) tiene dos equilibrios tumorales, una serie de simulaciones numéricas también muestran que el retraso ralentiza la convergencia de la solución a P^* del modelo (2.6). La Figura (3.7) muestra las órbitas

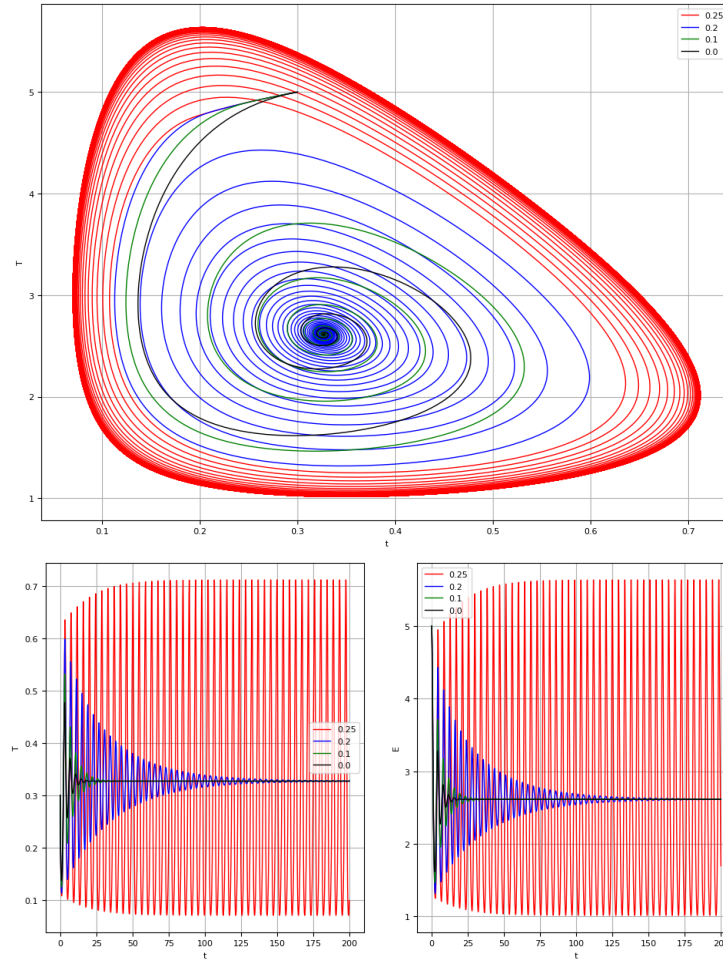


Figura 3.2. Las órbitas del modelo (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 0, 25, 0, 2, 0, 1, 0$ en el caso 1 de (2.1). Aquí, $r = 2, 5, K = 2, n = 0, 8, \sigma = 0, 5, \mu = 4, \eta = 1, 5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0, 3272, 2, 6138)$, y una órbita periódica estable.

del modelo (2.6) correspondiente a los diferentes valores de retraso τ , donde la columna de la izquierda es para $\tau = 0, 3$, la columna de la derecha es para $\tau = 2, 0$.

Las simulaciones numéricas anteriores muestran que el tiempo de retardo de la acción del tumor puede ralentizar la convergencia de las células tumorales al estado estacionario cuando el equilibrio del tumor P^* es estable, y puede aumentar la amplitud de la solución periódica como la solución periódica aparece. Consideramos el caso en que $\mu > 0$ y $n\sigma \geq r\eta$. La Figura (3.8) muestra el diagrama de fase del modelo (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ para tres valores de retardo de tiempo $\tau = 10, 5, 0$. Aquí, los parámetros del modelo son $r = 0, 5, K = 2, 0, n = 1, 8, \sigma = 2, 5, \mu = 4, 0, \eta = 1, 5$. Bajo estas condiciones, el modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 1, 667)$. Además, consideramos el caso en que $\mu < 0, \eta + \mu K < 0$, y $n\sigma > -\frac{rK}{4\mu} \left(\mu - \frac{\eta}{K}\right)^2$. En este escenario, la Figura (3.9) muestra

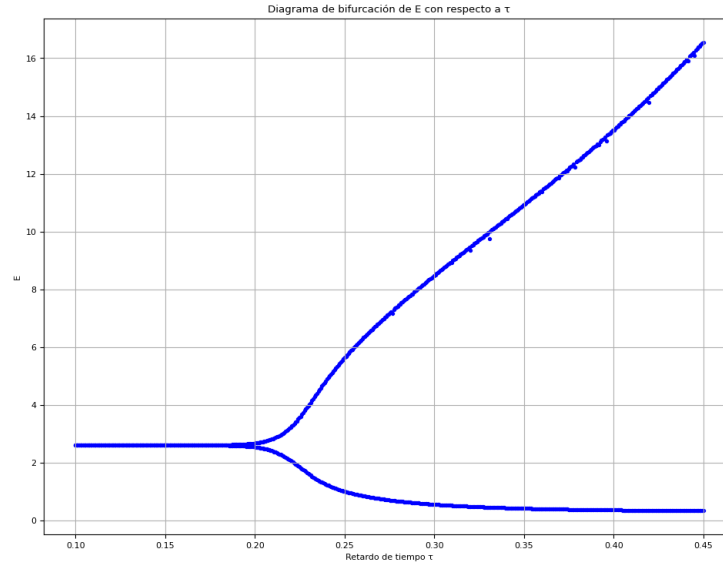


Figura 3.3. Diagrama de bifurcación del modelo (2.6) con respecto al retardo τ cuando $\mu K > \eta$ y $n\sigma < r\eta$. Aquí, excepto para el retardo τ , los valores de otros parámetros son los mismos que los de la Figura (3.1).

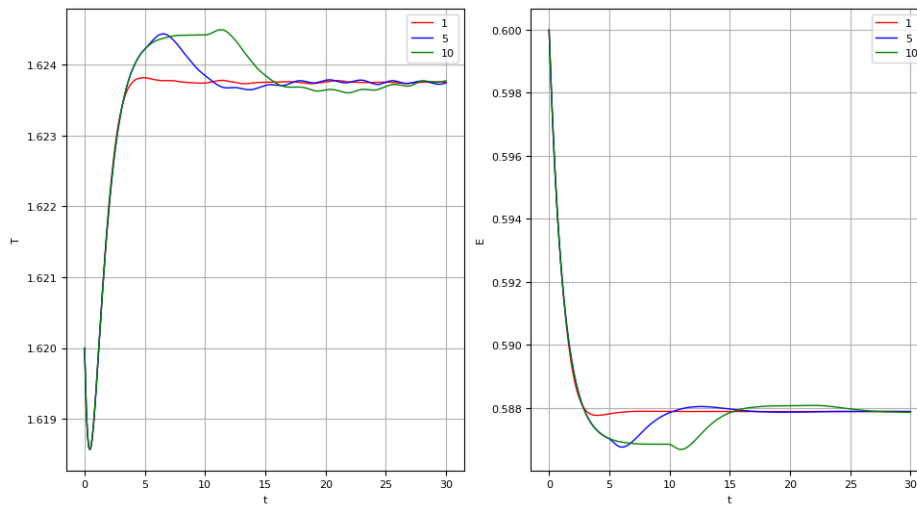


Figura 3.4. Las trayectorias de $T = T(t)$ y $E = E(t)$ del modelo (2.6) correspondientes a tres retardos $\tau = 1, 5, 10$ en caso de $0 < \mu < \frac{\eta}{K}$ y $n\sigma < r\eta$, respectivamente. Otros valores de los parámetros son $r = 2, 5, K = 2, n = 0, 8, \sigma = 0, 5, \mu = 0, 4$ y $\eta = 1, 5$.

las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$. Aquí, los valores de los parámetros son $r = 1, 5, K = 2, 0, n = 0, 9, \sigma = 0, 5, \mu = -2, 0, \eta = 0, 5$. Bajo estas condiciones, el modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 10)$. Finalmente, resumimos los resultados de las simulaciones en la Tabla (3.1), donde se presentan las configuraciones de parámetros utilizadas y los equilibrios tumorales correspondientes para cada caso.

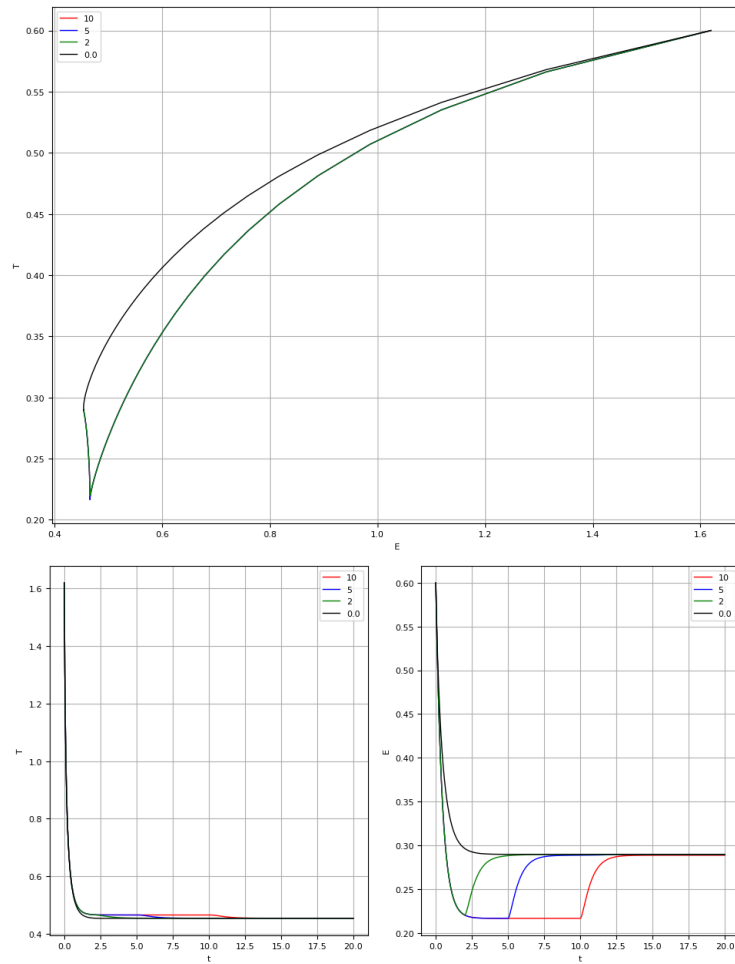


Figura 3.5. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$ en el caso $\mu < 0, \eta + \mu K \geq 0, n\sigma < r\eta$. Aquí, $r = 2, 5, K = 0, 5, n = 0, 8, \sigma = 0, 5, \mu = -0, 5, \eta = 1, 5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(0, 454, 0, 290)$, y una órbita periódica estable.

Caso	r	K	n	σ	μ	η	p_0	p^*	p_*	τ_0
1	2.5	2.0	0.8	0.5	4.0	1.5	[0.0, 0.333]	[0.327, 2.614]	-	0.224
2	2.5	0.5	0.8	0.5	-0.5	1.5	[0.0, 0.333]	[0.454, 0.290]	-	-
3	2.5	3.0	0.8	0.5	-2.5	1.5	[0.0, 0.333]	[2.946, 0.056]	-	-
4	2.5	4.0	0.8	6.2	-1.2	1.5	[0.0, 4.133]	[1.902, 1.639]	[0.848, 2.462]	-
6	0.5	2.0	1.8	2.5	4.0	1.5	[0.0, 1.667]	-	-	-
7	1.5	2.0	0.9	5.0	-2.0	0.5	[0.0, 10.0]	-	-	-

Tabla 3.1. Resumen de los resultados de las simulaciones basadas en los resultados documentados en la tabla (2.1).

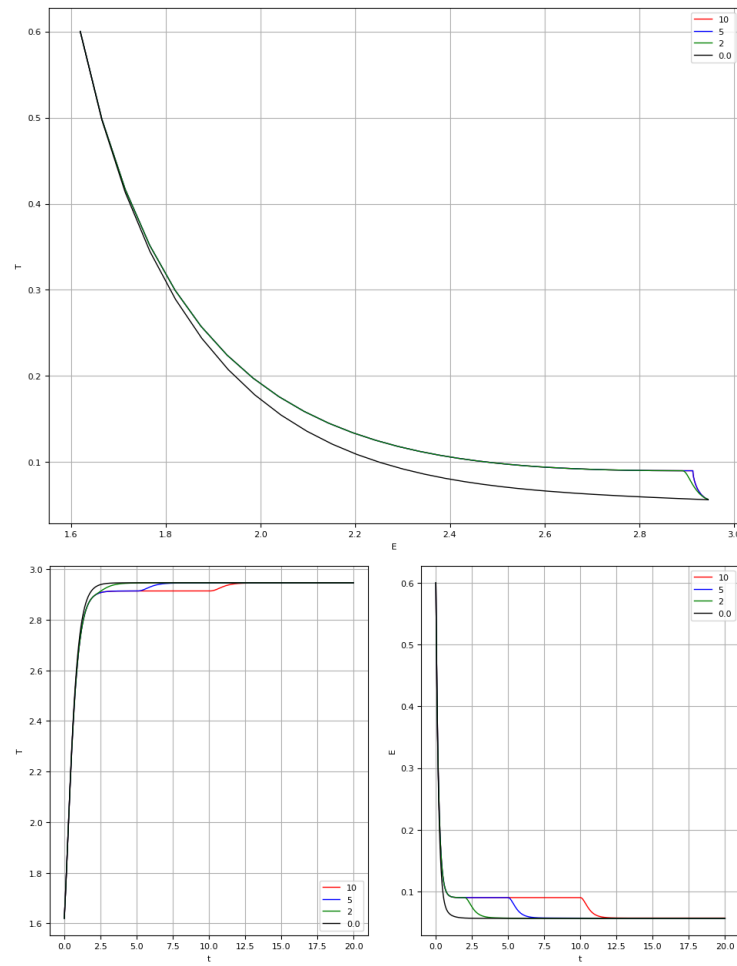


Figura 3.6. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$ en el caso $\mu < 0, \eta + \mu K < 0, n\sigma \leq r\eta$. Aquí, $r = 2, 5, K = 3, 0, n = 0, 8, \sigma = 0, 5, \mu = -2, 5, \eta = 1, 5$. El modelo (2.6) tiene un único equilibrio tumoral $P^*(2,946, 0,056)$, y una órbita periódica estable.

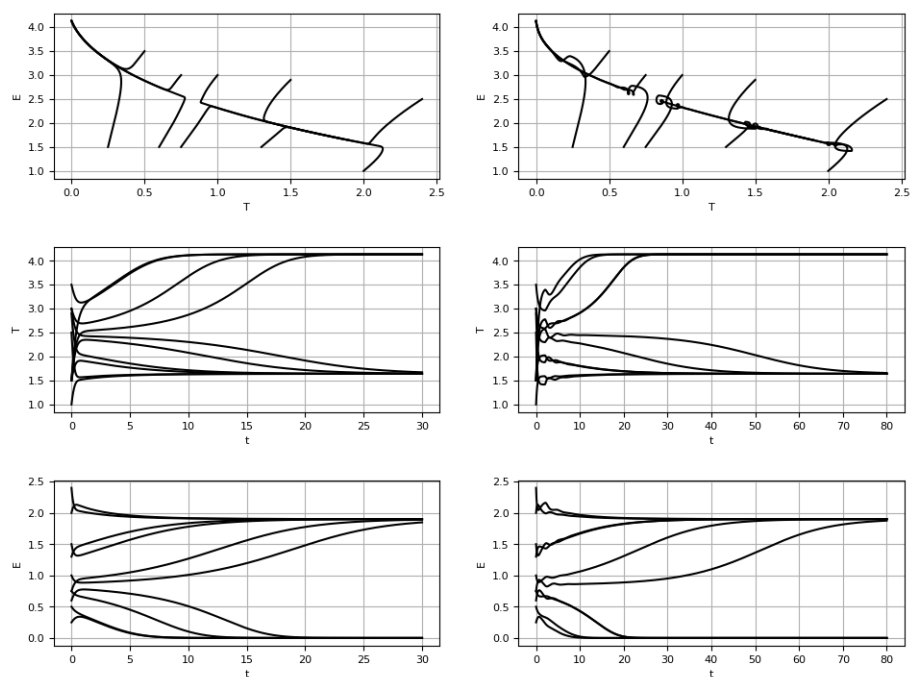


Figura 3.7. Las trayectorias del modelo (2.6) con dos retardos $\tau = 0.3, 2.0$ en el Caso C4 de la Tabla (2.1), donde la columna de la izquierda es para $\tau = 0.3$, la columna de la derecha es para $\tau = 2.0$, y las condiciones iniciales del modelo (2.6) son las mismas en las dos columnas. La primera fila son las órbitas del modelo (2.6) en el plano $T - E$, la segunda las trayectorias de $T = T(t)$, y la tercera las de $E = E(t)$.

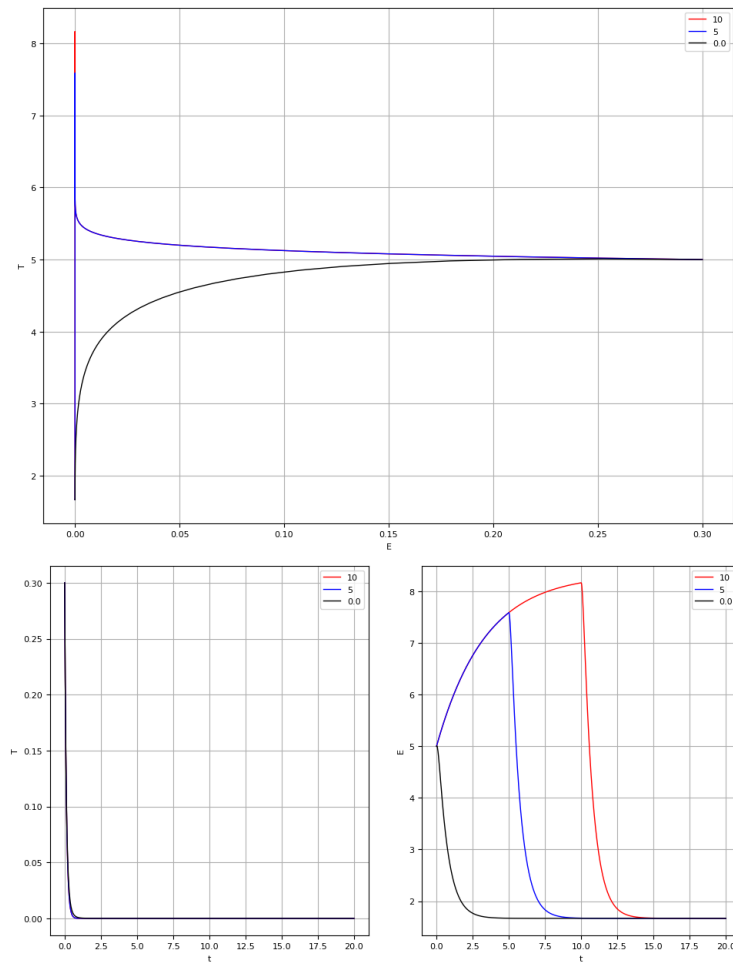


Figura 3.8. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a tres retardos de tiempo $\tau = 10, 5, 0$ en el caso $\mu > 0, n\sigma \geq r\eta$. Aquí, $r = 0,5, K = 2,0, n = 1,8, \sigma = 2,5, \mu = 4,0, \eta = 1,5$. El modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 1,667)$.

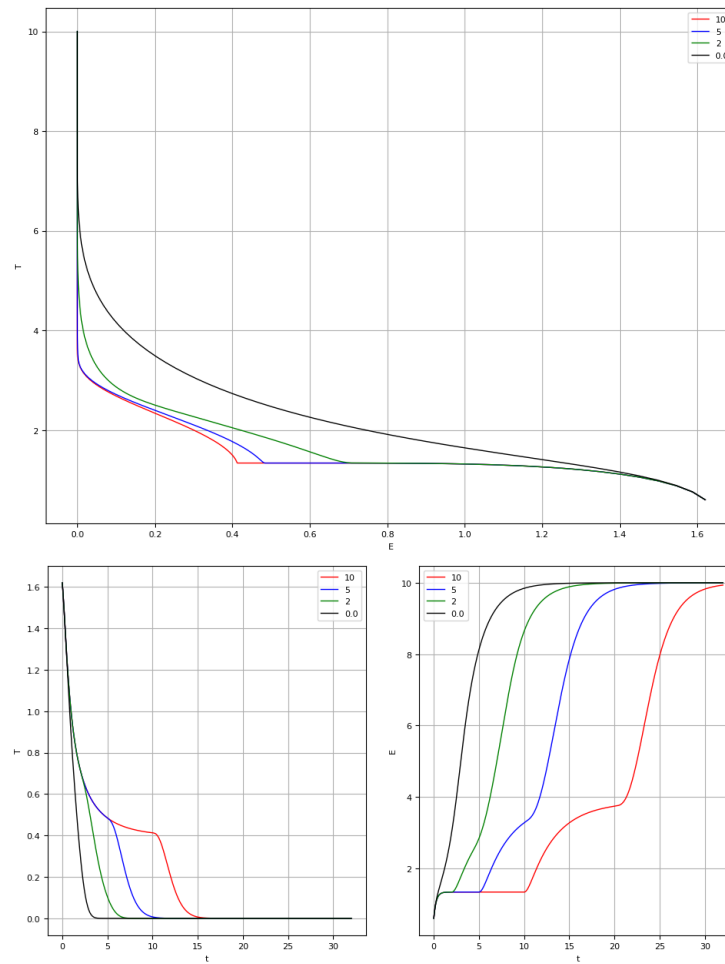


Figura 3.9. El diagrama de fase de (2.6) y las trayectorias de $T = T(t)$ y $E = E(t)$ correspondientes a cuatro retardos de tiempo $\tau = 10, 5, 2, 0$ en el caso $\mu < 0, \eta + \mu K < 0, n\sigma > -\frac{rK}{4\mu}(\mu - \frac{\eta}{K})^2$. Aquí, $r = 1,5, K = 2,0, n = 0,9, \sigma = 0,5, \mu = -2,0, \eta = 0,5$. El modelo (2.6) tiene un único equilibrio tumoral $P_0(0, 10)$.

Capítulo 4

Introducción a las ecuaciones diferenciales neurales

Hasta ahora hemos presentado dos modelos matemáticos que continuaremos analizando en el resto del documento, los sistemas (2.2) y (2.6) que describen la interacción entre células tumorales y células efectoras (del sistema inmunitario) en donde (2.2) es el modelo de crecimiento tumoral sin retardo temporal mientras que (2.6) es un modelo de crecimiento tumoral que posee un retardo de tiempo. Se aplicó la teoría clásica de sistemas dinámicos para obtener analíticamente información de las soluciones de equilibrio y de su estabilidad, y se llevaron a cabo simulaciones numéricas mostrando el aspecto general de las soluciones de estos dos modelos bajo diferentes valores de sus parámetros. Ahora, este capítulo introduce el concepto de ecuaciones diferenciales neuronales (NDEs), que combinan redes neuronales y ecuaciones diferenciales para modelar sistemas dinámicos complejos, para posteriormente aplicarlas a los modelos (2.2) y (2.6). El modelo matemático y las simulaciones numéricas de los capítulos anteriores proporcionan una base sólida para la transición al uso de ecuaciones diferenciales neuronales pero primero se hace necesario comprender que son y como se diseñan las NDEs.

En la intersección entre los sistemas dinámicos y el aprendizaje profundo, las ecuaciones diferenciales neuronales (NDEs) surgen como un puente entre dos disciplinas aparentemente dispares. Las NDEs, al demostrar que las ecuaciones diferenciales tradicionales son casos especiales parametrizados, ofrecen un enfoque poderoso y versátil. Desde la discretización de arquitecturas populares como las redes residuales hasta su aplicación en problemas generativos, sistemas dinámicos y series temporales, las NDEs se han posicionado como un área de reciente aparición y de gran interés tanto para la comunidad de aprendizaje automático moderno como para la modelización matemática tradicional. En esta sección, vamos a ver una

introducción a las ecuaciones diferenciales neuronales y nos vamos a basar principalmente en el trabajo de T. Q. Chen et al, del 2018 [22], que es considerado el artículo que convirtió a las NDEs en un área de investigación, y en la tesis doctoral de P. Kidger [6] del 2022, el mejor compendio de esta temática hasta el momento, se pueden ver en las fechas de estas referencias, este tema es increíblemente reciente, y está siendo desarrollado activamente hoy en día.

Definición 4.1. Una *ecuación diferencial neuronal* es una ecuación diferencial que utiliza una red neuronal para parametrizar el campo vectorial.

El ejemplo canónico de una **ecuación diferencial ordinaria neuronal** [6] es

$$\begin{aligned} y(0) &= y_0, \\ y'(t) &= f_\theta(t, y(t)), \end{aligned} \tag{4.1}$$

donde $y_0 \in \mathbf{R}^{d_1 \times \dots \times d_k}$ es un tensor de cualquier dimensión, θ representa algún vector de parámetros aprendidos y $f_\theta : \mathbf{R} \times \mathbf{R}^{d_1 \times \dots \times d_k} \rightarrow \mathbf{R}^{d_1 \times d_k}$ es una red neuronal. Generalmente, f_θ va a ser una arquitectura neuronal simple, como un perceptron multicapa o una red residual.

La primera pregunta que podemos llegar a hacernos es sobre la existencia y unicidad de una solución a la ecuación (4.1), y la respuesta es "sencilla", siempre que f_θ sea Lipschitz, algo que suele ser cierto en una red neuronal, generalmente estas son una composición de funciones Lipschitz entonces se aplica el teorema de existencia y unicidad de Picard [5]:

Teorema 4.2. Sea $f : [0, T] \times \mathbf{R}^d \rightarrow \mathbf{R}^d$ continua en t y uniformemente Lipschitz en y . Sea $y_0 \in \mathbf{R}^d$. Entonces existe una solución única y tal que $y : [0, T] \rightarrow \mathbf{R}^d$ satisfice:

$$\begin{aligned} y(0) &= y_0 \\ \frac{dy}{dt}(t) &= f(t, y(t)) \end{aligned}$$

Las NDEs pueden ser vistas como una extensión natural de las redes residuales (ResNets). Mientras que las ResNets utilizan sumas residuales discretas para pasar información entre capas, las NDEs adoptan un enfoque continuo, donde el paso de información se modela mediante la solución de una ecuación diferencial ordinaria (ODE). Este enfoque continuo permite una mayor flexibilidad y capacidad de modelado, especialmente para datos y procesos que son inherentemente continuos en el tiempo. Además, esta continuidad facilita la incorporación de conocimientos previos sobre la dinámica subyacente del sistema modelado, lo que puede mejorar tanto la precisión como la interpretabilidad de los modelos resultantes.

4.1. Redes neuronales de profundidad continua:

Modelos como las redes residuales, las redes neuronales recurrentes y los flujos normalizadores construyen transformaciones complicadas componiendo una secuencia de transformaciones a un estado oculto:

$$h_{t+1} = h_t + f_{\theta}(h_t, t),$$

donde $t \in \{0, \dots, T\}$ y $h_t \in \mathbf{R}^D$, es decir, los parámetros de todos los bloques se concatenan en θ . Pero, ¿qué ocurre a medida que añadimos más capas y damos pasos más pequeños? En el límite, parametrizamos la dinámica continua de las unidades ocultas mediante una ecuación diferencial ordinaria especificada por una red neuronal:

$$\frac{dh(t)}{dt} = f_{\theta}(h(t), t).$$

Discretizar esto utilizando el método de Euler explícito en tiempos t uniformemente separados por Δt nos da:

$$h_{t+1} - h(t) \approx \Delta t \frac{dh_t}{dt} = f_{\theta}(h_t, t),$$

de manera que

$$h_{t+1} = h_t + \Delta t f_{\theta}(h_t, t).$$

Absorbiendo Δt en f_{θ} , recuperamos la formulación de la primera ecuación. Luego, partiendo de la capa de entrada $h(0)$, podemos definir la capa de salida $h(T)$ como la solución a este problema de valor inicial de la EDO en un tiempo T . Este valor puede calcularse mediante un solucionador de ecuaciones diferenciales de caja negra, que evalúa la dinámica f de la unidad oculta siempre que sea necesario para determinar la solución con la precisión deseada. En la figura se contrastan estos dos enfoques.

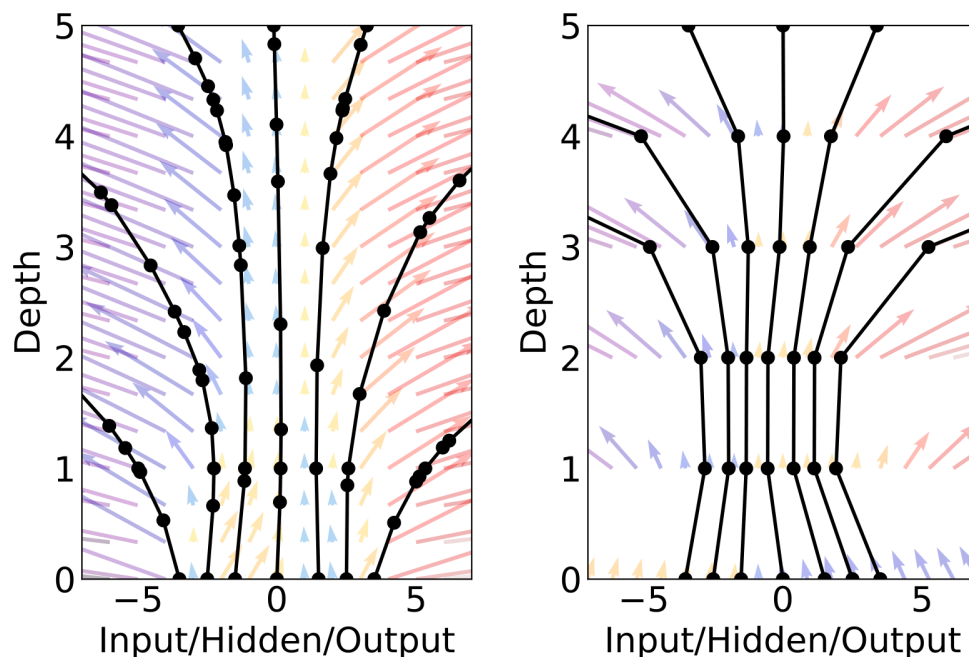


Figura 4.1. Izquierda: Una red EDO define un campo vectorial que transforma continuamente el estado. Los círculos representan lugares de evaluación. Derecha: Una red residual define una secuencia discreta de transformaciones finitas. Las imágenes fueron tomadas de un repositorio de [GitHub](#) del autor de [6].

Habiendo observado que las ecuaciones diferenciales neuronales son el límite continuo de las redes residuales, podemos empezar a hacer otras conexiones. Resulta que las características clave de una GRU o una LSTM, en comparación con las redes recurrentes genéricas y son reglas de actualización que se asemejan sospechosamente a ecuaciones diferenciales discretizadas. Las capas de acoplamiento en redes neuronales invertibles resultan estar relacionadas con solucionadores de ecuaciones diferenciales reversibles. Y así sucesivamente.

Por coincidencia (o, a medida que la idea se vuelve más popular, por diseño), muchas de las arquitecturas de aprendizaje profundo más efectivas y populares se asemejan a ecuaciones diferenciales. Quizás no deberíamos sorprendernos: las ecuaciones diferenciales han sido el paradigma dominante de modelado durante siglos; no son tan fáciles de derrocar.

4.2. Entrenamiento de las NDEs

El entrenamiento de las ecuaciones diferenciales neuronales (NDEs) es un proceso complejo que combina técnicas avanzadas de diferenciación automática, optimización y resolución de ecuaciones diferenciales. A continuación, se detallan los métodos y enfoques utilizados, incluyendo el uso de solucionadores especí-

ficos y optimizadores avanzados que se detallan en el capítulo 5 de [22].

4.2.1. Retropropagando a través del solucionador de ODEs

Para entrenar una NDE, es fundamental calcular los gradientes de los parámetros del modelo respecto a una función de pérdida que evalúa la calidad de las predicciones del modelo. Esto se logra mediante la retropropagación a través del solucionador de ecuaciones diferenciales. El proceso general incluye los siguientes pasos:

1. **Formulación del problema:** Se define una ecuación diferencial ordinaria (ODE) donde el campo vectorial f_θ está parametrizado por una red neuronal con parámetros θ . La ecuación se puede formular como:

$$\frac{dy}{dt} = f_\theta(t, y(t)), \quad y(0) = y_0$$

donde $y(t)$ es la variable de estado y f_θ es la red neuronal que modela las dinámicas del sistema.

2. **Solución de la ODE:** Se utiliza un solucionador de ODE numérico para integrar la ecuación diferencial desde el tiempo inicial t_0 hasta el tiempo final T . Esto produce una trayectoria $y(t)$ que depende de los parámetros θ . Los solucionadores de ODE pueden ser de paso fijo (como el método de Euler) o de paso variable (como Runge-Kutta de cuarto orden).
3. **Evaluación de la función de pérdida:** Se define una función de pérdida \mathcal{L} que mide la discrepancia entre las predicciones del modelo y los datos observados. Por ejemplo, en un problema de regresión, la función de pérdida podría ser el error cuadrático medio entre la trayectoria $y(t)$ y los datos observados $y_{\text{obs}}(t)$:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \|y(t_i) - y_{\text{obs}}(t_i)\|^2.$$

4. **Cálculo de gradientes:** Para actualizar los parámetros θ , es necesario calcular los gradientes de la función de pérdida respecto a θ . Esto se realiza mediante diferenciación automática, que se integra con el solucionador de ODE. El grafo computacional del solucionador de ODE se utiliza para computar las derivadas parciales necesarias.
5. **Actualización de parámetros:** Se utilizan métodos de optimización como el descenso de gradiente o variantes más sofisticadas (como Adam [25]) para actualizar los parámetros θ en la dirección opuesta

a los gradientes calculados, minimizando así la función de pérdida

$$\theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}(\theta)$$

donde η es la tasa de aprendizaje.

4.2.2. Método de la adjunta

El método de la adjunta es una técnica eficiente para calcular gradientes en NDEs, especialmente útil cuando las ecuaciones diferenciales son complejas o de alta dimensionalidad. Este método se basa en la teoría del cálculo de variaciones y proporciona una forma de retropropagar a través del solucionador de ODE sin necesidad de almacenar todas las derivadas intermedias. El proceso se describe a continuación:

1. **Definición de la ecuación adjunta:** Se define una ecuación diferencial adjunta que relaciona los gradientes de la función de pérdida con respecto a los estados $y(t)$ y los parámetros θ . La ecuación adjunta para una ODE dada es:

$$\frac{d\lambda}{dt} = -\lambda \frac{\partial f_{\theta}}{\partial y}, \quad \lambda(T) = \frac{\partial \mathcal{L}}{\partial y(T)}$$

donde $\lambda(t)$ es el multiplicador de Lagrange o variable adjunta.

2. **Solución de la ecuación adjunta:** La ecuación adjunta se integra hacia atrás en el tiempo desde T hasta t_0 , utilizando condiciones finales derivadas de la función de pérdida. Esto produce la trayectoria $\lambda(t)$.
3. **Cálculo del gradiente respecto a los parámetros:** Una vez que se tiene la solución de la ecuación adjunta $\lambda(t)$, se pueden calcular los gradientes de la función de pérdida respecto a los parámetros θ mediante:

$$\frac{d\mathcal{L}}{d\theta} = \int_{t_0}^T \lambda(t) \frac{\partial f_{\theta}}{\partial \theta} dt.$$

4. **Optimización:** Con los gradientes calculados, se procede a actualizar los parámetros θ utilizando un método de optimización como el descenso de gradiente.

El método de la adjunta es especialmente ventajoso porque evita la necesidad de almacenar toda la trayectoria $y(t)$ y sus derivadas intermedias, lo que reduce significativamente el uso de memoria y permite entrenar modelos más grandes y complejos.

4.2.3. Solucionador de ODE: Tsit5() y Controlador PID

En la práctica, el uso de solucionadores de ODE avanzados y controladores de precisión es crucial para la eficiencia y precisión del entrenamiento de NDEs. Dos componentes importantes son el solucionador Tsit5() y el controlador PID:

- **Tsit5():** Es un solucionador de ODE explícito de Runge-Kutta de orden 5, conocido por su eficiencia y precisión. Es especialmente útil para resolver problemas de ecuaciones diferenciales donde se requiere un equilibrio entre precisión y velocidad computacional. Su implementación en bibliotecas de aprendizaje automático permite una integración directa con modelos NDEs, facilitando el proceso de solución de ODEs durante el entrenamiento [23, 42].
- **Controlador PID:** `diffax.PIDController` es un controlador de paso adaptativo que ajusta dinámicamente el tamaño del paso del solucionador de ODE para mantener los errores de integración dentro de tolerancias específicas. Los parámetros `rtol` (tolerancia relativa) y `atol` (tolerancia absoluta) controlan la precisión de la solución. Un ejemplo de configuración podría ser

```
diffax.PIDController(rtol=1e-3, atol=1e-6),
```

lo que indica una alta precisión en la solución de la ODE. Este controlador ayuda a equilibrar la precisión y el costo computacional, adaptando el tamaño del paso según la dinámica del sistema, lo que es crucial para manejar sistemas con comportamientos altamente no lineales o estocásticos [23, 40].

4.2.4. Optimizador: `optax.adabelief`

El optimizador utilizado para actualizar los parámetros del modelo es crucial para la eficiencia y eficacia del entrenamiento. `optax.adabelief` es una variante avanzada del algoritmo de Adam, diseñada para mejorar la estabilidad y el rendimiento del entrenamiento:

- **AdaBelief:** Este optimizador se basa en el algoritmo Adam, pero introduce una corrección para el segundo momento de las actualizaciones de gradientes, haciendo que las actualizaciones sean más sensibles a los cambios repentinos en los gradientes. Esto resulta en una mejor convergencia y una mayor estabilidad durante el entrenamiento. La tasa de aprendizaje (`lr`) se puede ajustar según el problema específico [8, 49].

4.3. Potenciales aplicaciones

Las NDEs tienen una amplia gama de aplicaciones potenciales, incluyendo:

- **Aproximación universal en ED:** Las NDEs pueden aproximar soluciones de ecuaciones diferenciales arbitrarias [35], lo que las hace herramientas poderosas en el modelado de sistemas dinámicos complejos. Esto es particularmente relevante en campos donde las ecuaciones diferenciales describen fenómenos fundamentales, como la física, la biología y la ingeniería.
- **Aplicaciones a la física:** En física, las NDEs pueden utilizarse para modelar sistemas dinámicos continuos [44], como el movimiento de partículas bajo fuerzas complejas o la evolución de sistemas cuánticos. La capacidad de las NDEs para manejar dinámicas no lineales y complejas las hace especialmente útiles en este contexto.
- **Series temporales irregulares:** Las NDEs son particularmente útiles para manejar datos de series temporales que no están regularmente espaciados, permitiendo una mejor modelización y predicción en campos como la finanzas y la biomedicina. Esto se debe a que las NDEs pueden capturar de manera más precisa las dinámicas subyacentes en datos que no se ajustan a un intervalo de tiempo uniforme, mejorando así la calidad de las predicciones y análisis.

4.3.1. Ventajas

- **Alta capacidad de aproximación:** Las NDEs pueden modelar funciones altamente complejas y no lineales, lo que les permite capturar patrones y dinámicas que serían difíciles de modelar con técnicas tradicionales.
- **Flexibilidad:** Pueden manejar datos irregulares y dinámicos, adaptándose a una amplia variedad de problemas y contextos.
- **Eficiencia de memoria:** Utilizan la memoria de manera eficiente durante el entrenamiento, especialmente cuando se emplean técnicas como el método de la adjunta.
- **Teoría robusta:** Se apoyan en una sólida base teórica tanto del aprendizaje profundo como de los sistemas dinámicos, proporcionando una estructura matemática bien fundamentada para su análisis y aplicación.

4.3.2. Desventajas

- **Complejidad computacional:** El entrenamiento de NDEs puede ser computacionalmente intensivo, especialmente para grandes conjuntos de datos o modelos muy complejos. La necesidad de resolver ecuaciones diferenciales durante el entrenamiento añade una capa adicional de complejidad.
- **Sensibilidad a la parametrización:** La elección de los parámetros y la estructura de la red pueden tener un impacto significativo en el rendimiento y la estabilidad del modelo. Encontrar la parametrización óptima puede requerir un proceso de prueba y error y una comprensión profunda del problema específico.
- **Dificultad de interpretación:** Aunque las NDEs ofrecen una gran capacidad de modelado, interpretar los modelos resultantes puede ser más complejo en comparación con otros enfoques, debido a la naturaleza no lineal y a menudo opaca de las redes neuronales utilizadas para parametrizar el campo vectorial.

Las ecuaciones diferenciales neuronales representan una intersección fascinante entre el aprendizaje profundo y los sistemas dinámicos, ofreciendo nuevas herramientas y enfoques para una variedad de problemas complejos y dinámicos en diversas disciplinas.

Capítulo 5

Implementación del Modelo de Crecimiento Tumoral con NeuralODEs

Ahora vamos a entrar la implementación de los modelos de crecimiento tumorales (2.2) y (2.6), los enlaces a estas implementaciones se encuentran en el apéndice C, para esto nos guiamos de los ejemplos de la documentación de `diffax` y `equinox` [23, 24] Las NDEs se aplican para mejorar la precisión y la capacidad de generalización del modelo de crecimiento tumoral. Este enfoque permite capturar dinámicas más complejas y hacer predicciones más precisas sobre la evolución del tumor y la respuesta inmunitaria, integrando la flexibilidad de las redes neuronales con el rigor de las ecuaciones diferenciales. Específicamente, investigamos:

1. ¿Puede una NDE resolver el sistema de ecuaciones en una condición inicial nueva y aleatoria?
2. ¿Puede una red más simple, como una red neuronal recurrente (RNN por sus siglas en ingles), resolver el mismo problema? ¿Es necesario usar NDEs para este propósito?

La motivación principal detrás de este estudio es evaluar la eficiencia y precisión de las NDEs y realizar una comparación con las RNNs tradicionales, en la resolución de ecuaciones diferenciales que modelan procesos complejos como el crecimiento tumoral. Las ecuaciones diferenciales son fundamentales en la biología matemática, y encontrar métodos numéricos eficientes y precisos para resolverlas es crucial para el avance de la investigación en esta área.

5.1. Implementación con datos sin retardo

En primer lugar, vamos a realizar la implementación para (2.2) para esto vamos a generar datos sintéticos con base en las ecuaciones del sistema usando diferentes parámetros y condiciones iniciales aleatorias, basándonos en los resultados de los capítulos (2) y (3), luego con estos datos vamos a realizar el entrenamiento de una NeuralODE y una RNN y finalmente probar su desempeño con un nuevo conjunto de datos.

5.1.1. Generación de Datos

Para evaluar el rendimiento de las NeuralODEs y las RNNs, se requiere un conjunto de datos representativo de las dinámicas del modelo tumoral. Los datos se generaron simulando un sistema de ecuaciones diferenciales que modela el crecimiento tumoral. A continuación, se describe el proceso de generación de datos. Primero que todo resolvemos numéricamente el sistema de ecuaciones diferenciales para una variedad de condiciones iniciales y parámetros del modelo. Utilizamos el método de Runge-Kutta de quinto orden, más específicamente `Tsit5()`, para integrar el sistema de ecuaciones en el tiempo.

A continuación, se presenta el fragmento de código que generamos en Python para obtener la simulación del sistema de ecuaciones diferenciales:

```

1 def _get_data(ts, *, key):
2     # Se genera una condición inicial aleatoria.
3     y0 = jr.uniform(key, (2,), minval = jnp.array([0,0.5]), maxval = 2)
4     # Parametros del modelo.
5     r=2.5
6     K=2.
7     n=0.8
8     omega=0.5
9     miu=0.4
10    nu=1.5
11    def f(t, y, args):
12        # definición del sistema tumoral
13        T = y[0]
14        E = y[1]
15        x0 = r*T*(1-(T/K))-(n*T*E)
16        x1 = omega+miu*T*E-nu*E
17        return jnp.stack([x0, x1], axis=-1)
18    #Solver que es un método Runge-Kutta de orden 5
19    solver = diffrax.Tsit5()
20    dt0 = 0.01

```

```

21     saveat = diffrax.SaveAt(ts=ts)
22     sol = diffrax.diffeqsolve(
23         diffrax.ODETerm(f), solver, ts[0], ts[-1], dt0, y0, saveat=saveat
24     )
25     ys = sol.ys
26     #Retorna la solución
27     return ys
28
29
30 def get_data(dataset_size, *, key):
31     ts = jnp.linspace(0, 10, 100)
32     # Se separa la llave aleatoria en el tamaño del conjunto de datos.
33     key = jr.split(key, dataset_size)
34     # Vectorización de la función _get_data.
35     ys = jax.vmap(lambda key: _get_data(ts, key=key))(key)
36     return ts, ys

```

Código 5.1: La función `_get_data` genera una curva del modelo 2.2 con una condición inicial aleatoria mientras que `get_data` usa la función `_get_data` para generar un conjunto de curvas de tamaño `dataset_size` para el modelo.

5.1.2. Arquitectura de la Red

A continuación presentamos la arquitectura de la red que construimos basándonos en el ejemplo de NeuralODE de la documentación de diffrax [23], la cual está compuesta de dos clases `Func` que es un perceptron multicapa que funciona como insumo para la clase `NeuralODE`.

```

1  # Importar módulos necesarios de Equinox y Diffrax
2  import equinox as eqx
3  import diffrax
4  import jax.nn as jnn
5
6  # Definición de la clase Func que hereda de eqx.Module
7  class Func(eqx.Module):
8      mlp: eqx.nn.MLP # Declaración del atributo mlp como una red neuronal multi-capas (MLP)
9
10     def __init__(self, data_size, width_size, depth, *, key, **kwargs):
11         super().__init__(**kwargs) # Inicialización de la clase base eqx.Module

```

```

12     # Inicialización de la MLP con los parámetros dados
13     self.mlp = eqx.nn.MLP(
14         in_size=data_size,      # Tamaño de entrada
15         out_size=data_size,     # Tamaño de salida
16         width_size=width_size,  # Tamaño de las capas ocultas
17         depth=depth,            # Profundidad de la red (número de capas ocultas)
18         activation=jnn.softplus, # Función de activación a utilizar
19         key=key,                # Clave aleatoria para inicializar los pesos
20     )
21
22     # Método __call__ que define cómo se usa la clase como una función
23     def __call__(self, t, y, args):
24         return self.mlp(y) # Llamada a la MLP con la entrada y
25
26 # Definición de la clase NeuralODE que también hereda de eqx.Module
27 class NeuralODE(eqx.Module):
28     func: Func # Declaración del atributo func como una instancia de la clase Func
29
30     def __init__(self, data_size, width_size, depth, *, key, **kwargs):
31         super().__init__(**kwargs) # Inicialización de la clase base eqx.Module
32         # Inicialización de func como una instancia de Func con los parámetros dados
33         self.func = Func(data_size, width_size, depth, key=key)
34
35     # Método __call__ que define cómo se usa la clase como una función
36     def __call__(self, ts, y0):
37         # Resolver la ODE usando diffrax.diffeqsolve
38         solution = diffrax.diffeqsolve(
39             # Definición del término de la ODE usando self.func
40             diffrax.ODETerm(self.func),
41             # Método de integración (Tsitouras 5/4 Runge-Kutta)
42             diffrax.Tsit5(),
43             t0=ts[0],                # Tiempo inicial
44             t1=ts[-1],               # Tiempo final
45             dt0=ts[1] - ts[0],       # Paso de tiempo inicial
46             y0=y0,                   # Condiciones iniciales
47             # Controlador de tamaño de paso
48             stepsize_controller=diffrax.PIDController(rtol=1e-3, atol=1e-6),
49             saveat=diffrax.SaveAt(ts=ts), # Especificar cuándo guardar los resultados
50         )

```

```

51     # Devolver la solución de la ODE (valores en los tiempos especificados)
52     return solution.ys

```

Código 5.2: Funciones necesarias para la creación de la NeuralODE en donde Func crea un MLP y NeuralODE usa Func para definir una ODE

Ahora, vamos a explicar más en profundidad las características del código (5.2)

1. **Clase Func:** Define una red neuronal de tipo multi-capa (MLP) que será utilizada como la función dinámica dentro de una ODE (Ecuación Diferencial Ordinaria). Los componentes característicos de esta clase son:
 - **mlp:** Una instancia de una MLP de la librería Equinox.
 - **__init__:** El constructor de la clase, que inicializa la MLP con el tamaño de entrada, tamaño de salida, tamaño de las capas ocultas, profundidad de la red, función de activación y una clave aleatoria para inicialización.
 - **__call__:** Define cómo se utiliza la instancia de Func como una función que toma t , y , y $args$ (aunque t y $args$ no se usan directamente) y retorna el resultado de pasar y a través de la MLP.
2. **Clase NeuralODE:** Utiliza la instancia de Func para definir una ODE y resolverla usando el integrador de ecuaciones diferenciales de Diffraction. Los componentes característicos de esta clase son:
 - **func:** Una instancia de Func que representa la función dinámica de la ODE.
 - **__init__:** El constructor de la clase, que inicializa **func** con los parámetros necesarios.
 - **__call__:** Define cómo se utiliza la instancia de NeuralODE como una función que toma un array de tiempos ts y las condiciones iniciales y_0 , resuelve la ODE utilizando `diffraction.diffeqsolve`, y devuelve los valores de la solución en los tiempos especificados.

Con las características detalladas vamos a pasar a la explicación detallada del proceso de resolución de la ODE en el código 5.2

1. **Inicialización de la función dinámica (Func)** La clase Func se configura con una red neuronal que toma datos de entrada y , los procesa a través de varias capas ocultas, y produce una salida del mismo tamaño que la entrada. Esta red representa la derivada de y con respecto al tiempo t .
2. **Definición de la ODE (NeuralODE)** La clase NeuralODE utiliza una instancia de Func para definir la función dinámica de una ODE. Esta instancia (**func**) se pasa al integrador de ecuaciones diferenciales de Diffraction (`diffraction.ODETerm(self.func)`).

3. **Resolución de la ODE** El método `__call__` de NeuralODE usa `diffax.diffeqsolve` para resolver la ODE desde un tiempo inicial t_0 hasta un tiempo final t_1 , con un paso de tiempo inicial dt_0 y las condiciones iniciales y_0 . El integrador utilizado es `diffax.Tsit5()`, un método de Runge-Kutta de orden 5/4. Se usa un controlador PID para ajustar el tamaño de los pasos y se especifican los tiempos en los que se deben guardar los resultados.
4. **Resultado** La solución de la ODE, es decir, los valores de y en los tiempos especificados, se devuelve como salida.

5.1.3. Entrenamiento de la NeuralODE

El entrenamiento de la NeuralODE se llevó a cabo utilizando el optimizador Adam [25] y un enfoque de descenso de gradiente para minimizar el error cuadrático medio (MSE) entre las predicciones del modelo y los datos reales. A continuación, se muestra un fragmento de código que ilustra el proceso de entrenamiento.

```

1  # Se inicializa el modelo NeuralODE con los parámetros dados
2  model = NeuralODE(data_size, width_size, depth, key=model_key)
3
4  # Definición de la función de pérdida y su gradiente
5  @eqx.filter_value_and_grad
6  def grad_loss(model, ti, yi):
7      y_pred = jax.vmap(model, in_axes=(None, 0))(ti, yi[:, 0, :])
8      return jnp.mean((yi - y_pred) ** 2)
9
10 # Función para realizar un paso de optimización
11 @eqx.filter_jit
12 def make_step(ti, yi, model, opt_state):
13     loss, grads = grad_loss(model, ti, yi)
14     updates, opt_state = optim.update(grads, opt_state)
15     model = eqx.apply_updates(model, updates)
16     return loss, model, opt_state
17
18 # Entrenamiento del modelo NeuralODE
19 for lr, steps, length in zip(lr_strategy, steps_strategy, length_strategy):
20     # Inicialización del optimizador
21     optim = optax.adabelief(lr)
22     opt_state = optim.init(eqx.filter(model, eqx.is_inexact_array))

```

```

23
24     # Seleccionar un subconjunto de datos para el entrenamiento
25     _ts = ts[: int(length_size * length)]
26     _ys = ys[:, : int(length_size * length)]
27
28     # Loop de entrenamiento
29     for step, (yi,) in zip(
30         range(steps), dataloader((_ys,), batch_size, key=loader_key)
31     ):
32         start = time.time()
33         loss, model, opt_state = make_step(_ts, yi, model, opt_state)
34         end = time.time()

```

Código 5.3: Código del entrenamiento de la neuralODE.

La NeuralODE es una arquitectura que combina redes neuronales con métodos numéricos para la resolución de ecuaciones diferenciales. En este trabajo, luego del entrenamiento (5.3) realizamos una búsqueda exhaustiva (*grid search*) para determinar la mejor arquitectura para la NeuralODE. La búsqueda incluyó variaciones en la cantidad de capas y el tamaño de cada capa. Se corrió el entrenamiento para diferentes semillas aleatorias y se calculó el error basado en un nuevo conjunto de datos con condiciones iniciales aleatorias.

1. Resultados de la Búsqueda Exhaustiva: Las siguientes imágenes muestran los resultados de la *grid search*:

La figura (5.1) presenta una visualización del error cuadrático medio (MSE) obtenido para distintas configuraciones de la NeuralODE. En esta figura, el eje x representa el tamaño de las capas ocultas (número de neuronas), mientras que el eje y representa el número de capas ocultas. Los colores indican el valor del MSE, con colores más oscuros representando errores menores.

La Figura (5.2) muestra el mismo análisis mucho más detallado en un boxplot para diferentes configuraciones de la NeuralODE. El eje x representa la profundidad o cantidad de capas, el eje y representa el error (MSE) y el tamaño de cada capa está indicado por el color.

A partir de los resultados de la búsqueda exhaustiva, se seleccionó una arquitectura con cuatro capas ocultas de 16 unidades cada una, la cual mostró el error más pequeño tras varias corridas con nuevos datos.

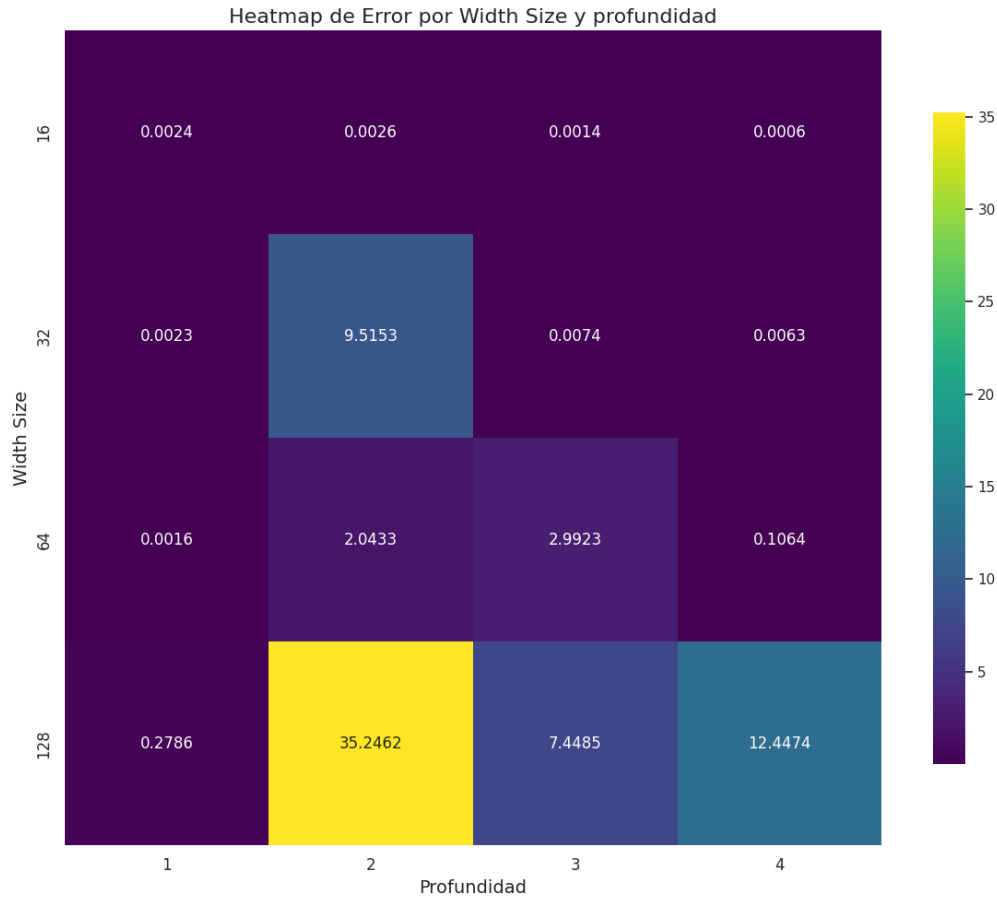


Figura 5.1. Resultados de la *grid search* para diferentes configuraciones de la NeuralODE. El eje x representa el tamaño de las capas, el eje y representa la cantidad de capas, y el color indica el MSE.

5.1.4. Resultados de la NeuralODE

Los resultados de la NeuralODE se comparan con los valores reales del sistema de ecuaciones. A continuación se presentan las gráficas que muestran la predicción del modelo frente a los valores reales

En la figura (5.3), se presentan las predicciones de la NeuralODE en comparación con los valores reales del sistema de ecuaciones. Las variables T y E representan diferentes componentes del sistema, y las gráficas muestran cómo estas variables evolucionan a lo largo del tiempo. La precisión del modelo se evalúa visualmente mediante la superposición de las predicciones y los datos reales.

La capacidad de la NeuralODE para capturar la dinámica del sistema de ecuaciones diferenciales se refleja en la estrecha concordancia entre las líneas de predicción y los datos reales, lo que sugiere que la NeuralODE es eficaz para este tipo de problemas.

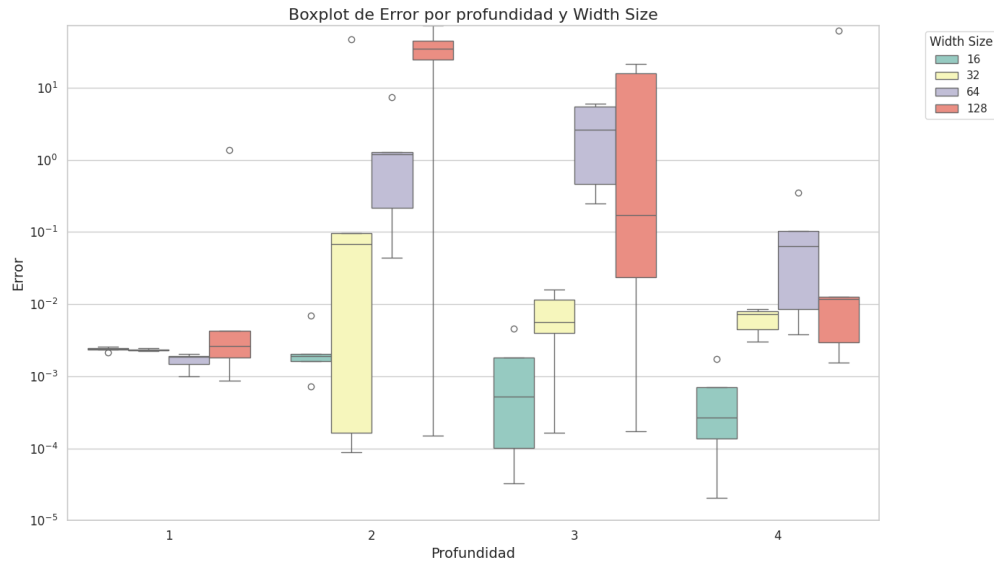


Figura 5.2. Resultados de la *grid search* para diferentes configuraciones de la NeuralODE. El eje x representa la profundidad o cantidad de capas, el eje y representa el error (MSE) y el tamaño de cada capa está indicado por el color.

5.1.5. Implementación de la RNN

Para comparar con la NeuralODE, se implementó una RNN guiándonos con el ejemplo de RNN en la documentación de equinox [24] utilizando una arquitectura con dos capas recurrentes y 64 unidades en cada capa. La elección de la compuerta recurrente fue una RNN simple y se utilizó el optimizador Adam la cual fue entrenada con los mismos datos a fin de responder la segunda pregunta, pueden consultar el Apéndice B. La definición de esta red se muestra a continuación para más información acerca de las redes neuronales recurrentes.

```

1 class RNN(eqz.Module):
2     hidden_size: int # Tamaño del estado oculto
3     cell: eqz.Module # Celda recurrente (GRU en este caso)
4     linear: eqz.nn.Linear # Capa lineal para la proyección de la salida
5     bias: jax.Array # Sesgo para la capa de salida
6     sequence_length: int # Longitud de la secuencia
7
8     def __init__(self, in_size, out_size, hidden_size, *, key, sequence_length=100):
9         # Dividir la clave aleatoria en dos para inicializar la celda GRU y la capa
10        lineal
11        ckey, lkey = jrandom.split(key)
12        # Inicializar el tamaño del estado oculto

```

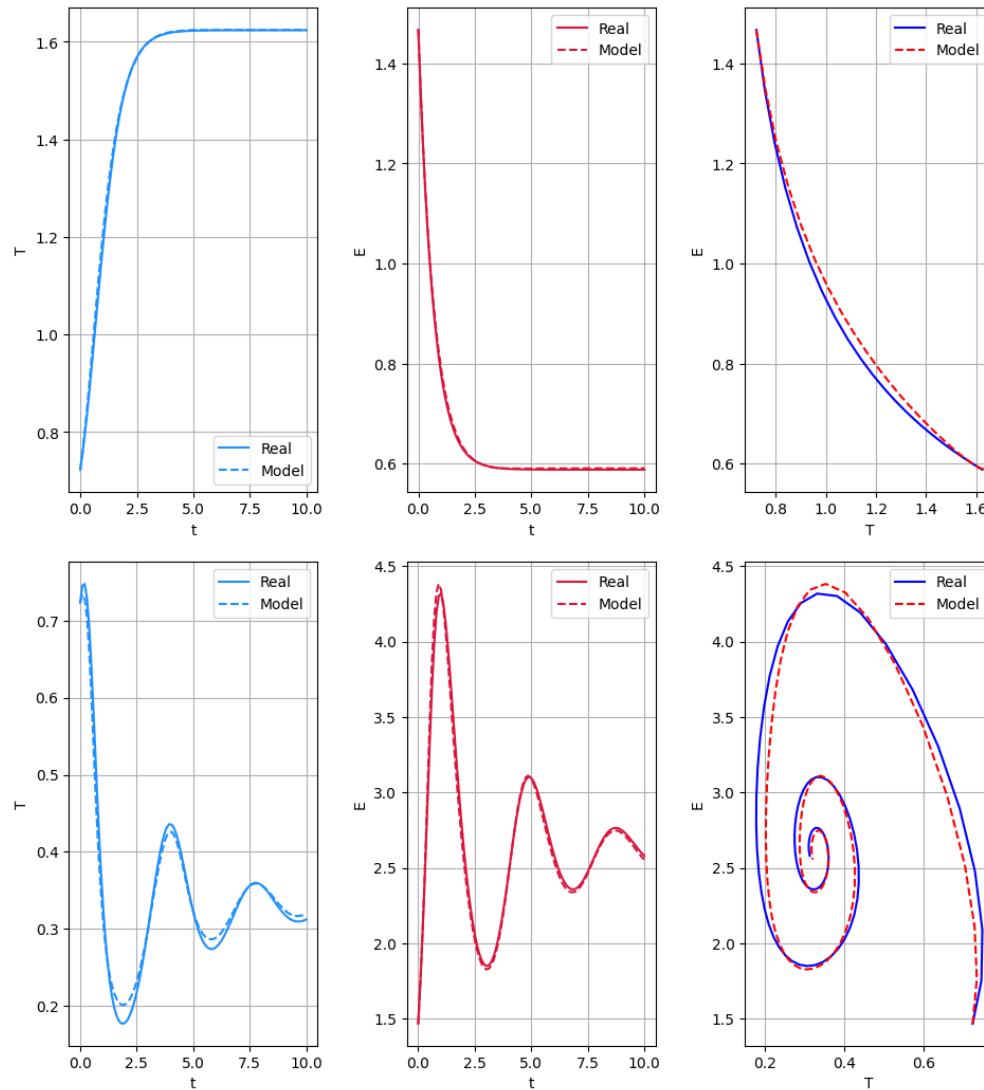


Figura 5.3. Comparación de las predicciones de la NeuralODE con los valores reales del sistema de ecuaciones que tiene condiciones iniciales completamente nuevas y aleatorias. Como podemos ver, el modelo se ajusta fielmente a la solución real.

```

13     self.hidden_size = hidden_size
14
15     # Inicializar una celda GRU con el tamaño de entrada y el tamaño oculto
16     self.cell = eqx.nn.GRUCell(in_size, hidden_size, key=ckey)
17
18     # Inicializar una capa lineal sin sesgo
19     self.linear = eqx.nn.Linear(hidden_size, out_size, use_bias=False, key=lkey)
20
21     # Inicializar el sesgo como un array de ceros con el tamaño de salida
22     self.bias = jnp.zeros(out_size)

```

```

23
24     # Establecer la longitud de la secuencia
25     self.sequence_length = sequence_length
26
27     def __call__(self, input):
28         # Inicializar el estado oculto como ceros
29         hidden = jnp.zeros((self.hidden_size,))
30
31         # Definir la función que se escaneará a través de la secuencia
32         def f(carry, _):
33             hidden, input = carry
34             # Actualizar el estado oculto usando la celda GRU
35             hidden = self.cell(input, hidden)
36             # Calcular la salida usando la capa lineal y sumar el sesgo
37             output = self.linear(hidden) + self.bias
38             # Retornar el estado oculto actualizado y la salida
39             return (hidden, output), output # carry, cosa para apilar
40
41         # Inicializar el estado con el estado oculto y la entrada
42         init_state = (hidden, input)
43
44         # Aplicar la función de escaneo a través de la longitud de la secuencia
45         (_, _), outs = lax.scan(f, init_state, None, length=self.sequence_length)
46
47         # Apilar las salidas a lo largo de una nueva dimensión y retornar
48         return jnp.stack(outs)

```

Código 5.4: Código del entrenamiento de la neuralODE

Clase RNN

Esta clase define una red neuronal recurrente simple utilizando Equinox (eqx), una biblioteca basada en JAX.

Atributos

- **hidden size:** Tamaño del estado oculto de la RNN.
- **cell:** Una celda GRU que actualiza el estado oculto.
- **linear:** Una capa lineal que proyecta el estado oculto al espacio de salida.
- **bias:** Un sesgo que se suma a la salida de la capa lineal.

- **sequence length:** Longitud de la secuencia para la RNN.

Método `__init__`

- Inicializa la RNN con los tamaños de entrada, salida y oculto, así como una clave aleatoria y la longitud de la secuencia.
- Divide la clave aleatoria en dos para inicializar la celda GRU y la capa lineal.
- Inicializa la celda GRU con el tamaño de entrada (*in_size*) y el tamaño oculto (*hidden_size*).
- Inicializa la capa lineal con el tamaño oculto y el tamaño de salida (*out_size*), sin utilizar un sesgo.
- Inicializa el sesgo como un array de ceros con el tamaño de salida.
- Establece la longitud de la secuencia.

Método `__call__`

- Inicializa el estado oculto como ceros.
- Define una función f que se escaneará a través de la secuencia:
 - Toma el estado oculto y la entrada como *carry*.
 - Actualiza el estado oculto usando la celda GRU.
 - Calcula la salida usando la capa lineal y le suma el sesgo.
 - Retorna el estado oculto actualizado y la salida.
- Inicializa el estado con el estado oculto y la entrada.
- Aplica la función de escaneo (*lax.scan*) a través de la longitud de la secuencia.
- Apila las salidas a lo largo de una nueva dimensión y las retorna.

5.1.6. Análisis de Resultados y de Errores

En la Figura (5.4), se presentan las predicciones de la RNN junto con los valores reales y los valores pronosticados por la NeuralODE. Las gráficas muestran que, aunque la RNN puede capturar algunas de las características principales del sistema, sus predicciones no son tan precisas como las de la NeuralODE. Esto se refleja en la mayor dispersión de las líneas de predicción en comparación con los datos reales.

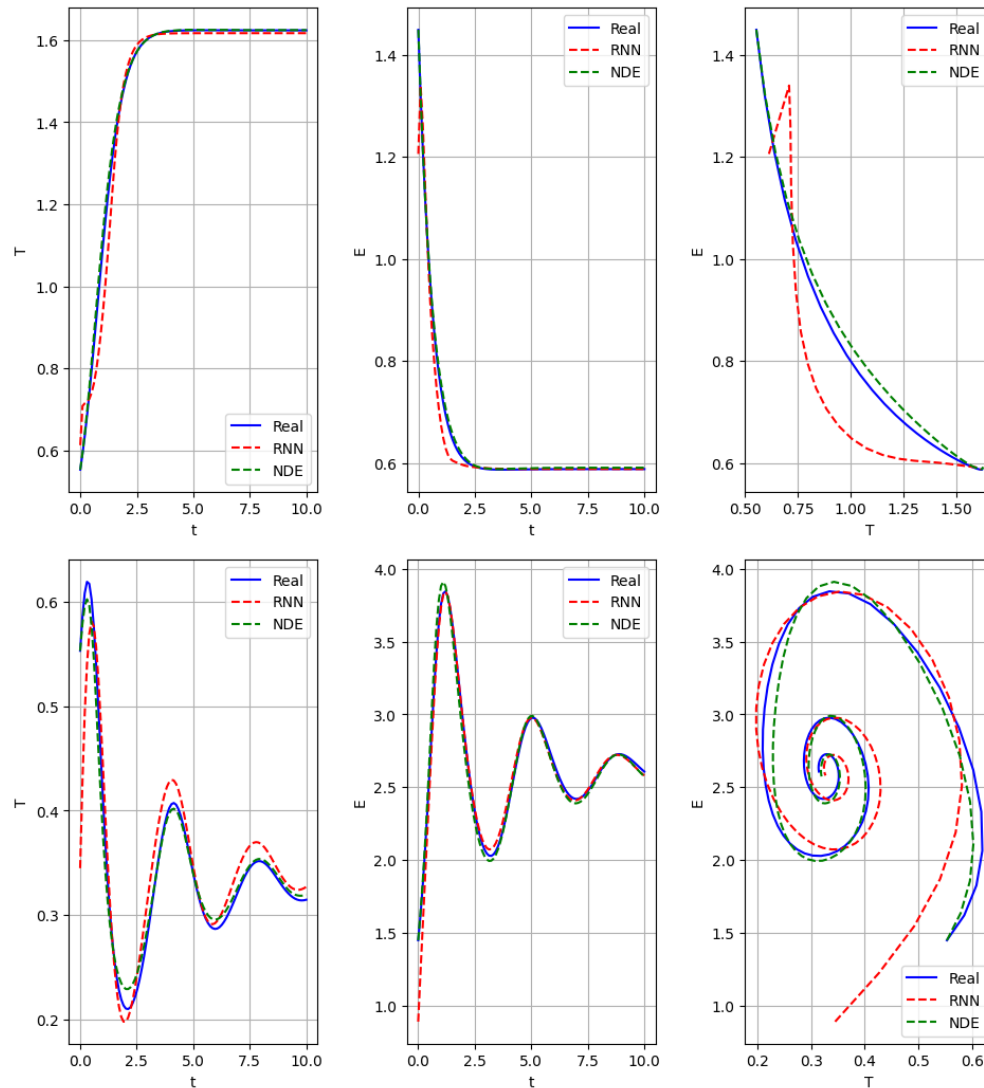


Figura 5.4. Comparación de las predicciones de la RNN junto con los valores reales y los valores pronosticados por la NeuralODE. El gráfico muestra la evolución de las variables T y E en el tiempo.

Para cuantificar el rendimiento de cada modelo, calculamos el error cuadrático medio (MSE) de las predicciones en comparación con los valores reales. Los errores obtenidos son los siguientes.

■ **RNN:**

- Error en T : 0.021728052
- Error en E : 0.010076428
- Error medio: 0.01590224

■ **NeuralODE:**

- Error en T: 0.005826596
- Error en E: 0.00060446496
- Error medio: 0.0032155307

La comparación de los errores MSE entre la NeuralODE y la RNN muestra que la NeuralODE es significativamente más precisa en la predicción de las variables T y E del sistema de ecuaciones diferenciales. La menor dispersión y los menores valores de MSE indican que la NeuralODE tiene una mayor capacidad para capturar la dinámica compleja del sistema.

1. **NeuralODE vs. RNN:** La NeuralODE demostró un mejor rendimiento en términos de precisión, como se observa en los menores valores de MSE. Esto confirma que la integración de métodos numéricos en la arquitectura de la red puede mejorar la capacidad de la red para resolver sistemas de ecuaciones diferenciales.
2. **Generalización:** Las NeuralODEs tienen una ventaja teórica en la resolución de sistemas de ecuaciones diferenciales debido a su integración con métodos numéricos, lo que puede resultar en una mejor generalización a condiciones iniciales nuevas y aleatorias. Esto se refleja en la menor dispersión de los errores en diferentes condiciones iniciales.
3. **Simplicidad y Eficiencia:** A pesar de que la RNN es una arquitectura más simple y puede ser más eficiente en términos de tiempo de entrenamiento y recursos computacionales, la NeuralODE mostró una mejor capacidad para capturar la dinámica compleja del sistema de ecuaciones diferenciales del modelo tumoral.
4. **Aplicaciones Prácticas:** En aplicaciones prácticas donde la precisión extrema y la capacidad de generalización son críticas, como en la modelización de sistemas biológicos complejos, las NeuralODEs pueden ser la opción preferida. Sin embargo, en situaciones donde los recursos computacionales son limitados y se puede tolerar una menor precisión, las RNNs pueden ser una alternativa viable.

5.2. Implementación con datos con retardo

Para esta sección vamos a generar un conjunto de datos que describa el comportamiento del modelo (2.6). Es decir, que incluya el retardo de tiempo y vamos a ver como funcionan las NeuralODE para este nuevo conjunto de datos.

5.2.1. Generación de Datos

Para generar datos de entrenamiento y validación, resolvemos numéricamente el sistema de ecuaciones diferenciales para una variedad de condiciones iniciales y parámetros del modelo. En este caso usaremos una librería de python llamada jitcdde la cual fue creada para resolver ecuaciones diferenciales con retardo.

A continuación, se presenta el fragmento de código que ilustra la simulación del sistema de ecuaciones diferenciales con retardo:

```

1 def get_data(num_initial_points):
2     # Definir constantes usadas en las ecuaciones
3     r = 2.5
4     K = 2.0
5     n = 0.8
6     omega = 0.5
7     miu = 4.0
8     nu = 1.5
9
10    # Definir variable simbólica para el retardo
11    d = symengine.symbols("d")
12
13    # Definir las ecuaciones diferenciales con retardo
14    equations = [
15        r * y(0, t) * (1 - (y(0, t) / K)) - n * y(1, t) * y(0, t), # Ecuación 1
16        omega + miu * y(0, t - d) * y(1, t) - nu * y(1, t) # Ecuación 2
17    ]
18
19    # Crear el sistema de ecuaciones diferenciales con retardo usando jitcdde
20    ddesys = jitcdde(equations, control_pars=[d], max_delay=0.2)
21
22    # Definir un rango de tiempo para la integración
23    ts = np.linspace(0, 30, 1000)
24
25    # Generar puntos iniciales aleatorios
26    np.random.seed(67) # Fijar la semilla para reproducibilidad
27    initial_points = np.random.uniform(0, 2, size=(num_initial_points, 2)) # Puntos
    iniciales en el rango [0, 2)
28
29    ys = [] # Lista para almacenar los resultados de la integración
30    for initial_point in initial_points:

```

```

31     # Establecer el estado pasado constante con los puntos iniciales
32     ddesys.constant_past(initial_point)
33     params = [0.2] # Definir los parámetros de control
34     ddesys.set_parameters(*params) # Establecer los parámetros del sistema
35
36     y_values = [] # Lista para almacenar los valores de y en cada tiempo t
37     for t_val in ts:
38         # Integrar el sistema para el tiempo t_val
39         y_values.append(ddesys.integrate(t_val))
40
41     ys.append(y_values) # Agregar los valores de y para el punto inicial actual a
42     la lista ys
43
44     ys = np.array(ys) # Convertir la lista ys a un array de NumPy
45     ts = jax.device_put(ts) # Transferir el array ts al dispositivo (usando JAX)
46     ys = jax.device_put(ys) # Transferir el array ys al dispositivo (usando JAX)
47
48     # Devolver los valores de y y los tiempos
49     return ys, ts

```

Código 5.5: Función para la generación de datos para el modelo.

Con este nuevo conjunto de datos volvemos a realizar el análisis de resultados de la NeuralODE y la RNN.

5.2.2. Análisis de Resultados y Comparación de Errores

En la Figura (5.5), se presentan las predicciones de la RNN junto con los valores reales con retardo y los valores pronosticados por la NeuralODE. Las gráficas muestran que, para este conjunto de datos, los resultados de ambos modelos no es el más óptimo, ninguno es capaz de reconocer los cambios en las trayectorias cuando se añade el retardo de tiempo.

Para cuantificar el rendimiento de cada modelo, calculamos el error cuadrático medio (MSE) de las predicciones en comparación con los valores reales con delay. Los errores obtenidos son los siguientes.

■ RNN:

- Error en T: 0.89506215
- Error en E: 0.017895624

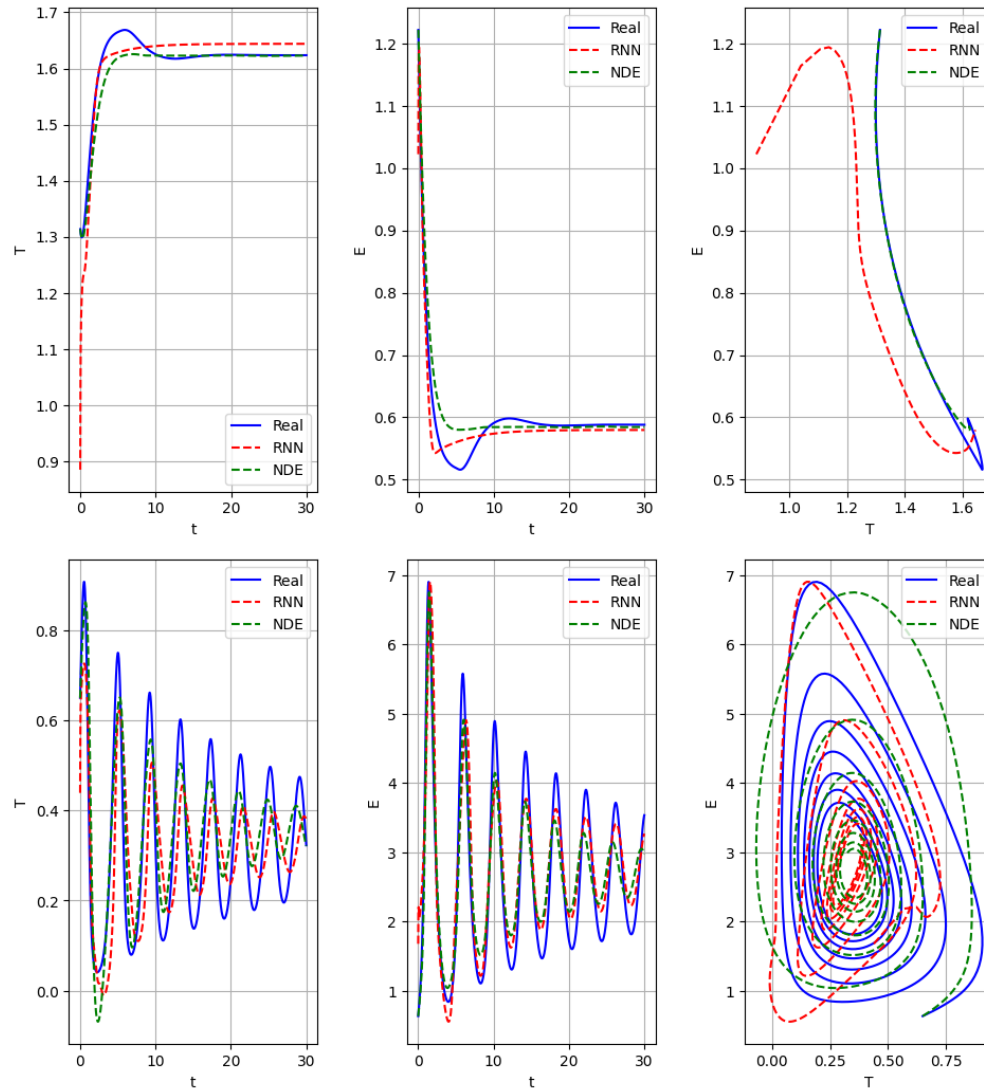


Figura 5.5. Comparación de las predicciones de la RNN junto con los valores reales con retardo y los valores pronosticados por la NeuralODE. El gráfico muestra la evolución de las variables T y E en el tiempo.

- Error medio: 0.4564789

■ **NeuralODE:**

- Error en T : 0.5015076
- Error en E : 0.030566195
- Error medio: 0.2660369

La comparación de los errores MSE entre la NeuralODE y la RNN muestra que la NeuralODE aunque sigue siendo más precisa en la predicción de las variables T y E del sistema de ecuaciones diferenciales. Ninguno

de los modelos usados tuvo un buen desempeño. Dados los anteriores resultados, se puede llegar a concluir que las NeuralODEs no son eficaces para modelos de ecuaciones diferenciales con retardo. Sin embargo, las ecuaciones diferenciales neuronales son una rama muy nueva del deep learning y actualmente existen trabajos con el de [47] en el que se plantean ecuaciones diferenciales neuronales con retardo o NDDEs por sus siglas en inglés, debido a su complejidad decidimos dejar el estudio estas para un trabajo posterior, pero podemos dar indicio de que proponen.

5.3. Introducción a las NDDEs

Las Neural Delay Differential Equations (NDDEs) representan una extensión significativa de las Neural Ordinary Differential Equations (NODEs) al incorporar efectos de retardo temporal en las ecuaciones diferenciales que gobiernan el sistema. Esto permite una capacidad de representación no lineal más robusta y versátil [47].

Las NDDEs se definen mediante un sistema de ecuaciones diferenciales con retardo (DDEs). La formulación básica es la siguiente:

$$\begin{cases} \frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), \mathbf{h}(t - \tau); \mathbf{w}), & t \geq 0 \\ \mathbf{h}(t) = \phi(t), & t \leq 0 \end{cases}$$

Aquí:

- $\mathbf{h}(t)$ es el estado del sistema en el tiempo t .
- f es la función que define la dinámica del sistema, dependiente del estado actual $\mathbf{h}(t)$ y del estado retrasado $\mathbf{h}(t - \tau)$.
- τ es el tiempo de retardo.
- \mathbf{w} son los parámetros del modelo a aprender.
- $\phi(t)$ es la función inicial que define el estado del sistema antes de $t = 0$.

5.3.1. Método de Sensibilidad Adjunto para las NDDEs

El método de sensibilidad adjunto es esencial para calcular los gradientes de la función de pérdida respecto a los parámetros del modelo de manera eficiente. Este método se adapta para las NDDEs de la siguiente forma:

Definición del Adjunto

La variable adjunta $\lambda(t)$ se define como $\lambda(t) = \frac{\partial L(\mathbf{h}(T))}{\partial \mathbf{h}(t)}$, donde L es la función de pérdida.

Dinámica del Adjunto

La dinámica del adjunto también sigue una DDE:

$$\begin{cases} \frac{d\lambda(t)}{dt} = -\lambda(t)^\top \frac{\partial f(\mathbf{h}(t), \mathbf{h}(t-\tau); \mathbf{w})}{\partial \mathbf{h}(t)} - \lambda(t+\tau)^\top \frac{\partial f(\mathbf{h}(t+\tau), \mathbf{h}(t); \mathbf{w})}{\partial \mathbf{h}(t)} \chi_{[0, T-\tau]}(t), & t \leq T \\ \lambda(T) = \frac{\partial L(\mathbf{h}(T))}{\partial \mathbf{h}(T)} \end{cases}$$

donde $\chi_{[0, T-\tau]}(t)$ es una función característica que es 1 si $t \in [0, T - \tau]$ y 0 en caso contrario.

Cálculo de los Gradientes

Los gradientes de la función de pérdida respecto a los parámetros \mathbf{w} se calculan como:

$$\frac{dL}{d\mathbf{w}} = \int_0^T -\lambda(t)^\top \frac{\partial f(\mathbf{h}(t), \mathbf{h}(t-\tau); \mathbf{w})}{\partial \mathbf{w}} dt$$

5.3.2. Algoritmo de Solución

El algoritmo propuesto para resolver las NDDEs y calcular los gradientes de manera eficiente es el siguiente:

Paso Adelante (Forward Pass)

Resolver la NDDE desde $t = 0$ hasta $t = T$ para obtener $\mathbf{h}(T)$.

Cálculo del Adjunto

Resolver la ecuación de adjunto hacia atrás en el tiempo desde $t = T$ hasta $t = 0$.

Cálculo de los Gradientes

Utilizar la solución del adjunto para integrar y obtener los gradientes de la función de pérdida respecto a los parámetros del modelo.

Este algoritmo permite que las NDDEs sean computacionalmente viables, aprovechando técnicas avanzadas de cálculo de gradientes y optimización.

Capítulo 6

Conclusiones

Se realizó un análisis profundo de los sistemas tumorales (2.2) y (2.6), donde se llevaron a cabo todos los cálculos necesarios para determinar la existencia y no negatividad de los puntos de equilibrio, concluyendo así la estabilidad de cada uno. Posteriormente, utilizando un módulo específico para ecuaciones diferenciales con retardo (`jitcdde`), realizamos una serie de simulaciones basándonos en las condiciones de la tabla (2.1), observando el comportamiento general del modelo bajo distintos parámetros, con esto construimos una base sólida para luego realizar la implementación con ecuaciones diferenciales neuronales.

Se logró un entendimiento profundo de la teoría básica de las ecuaciones diferenciales neuronales mediante revisión exhaustiva de literatura especializada. Se exploraron sus fundamentos matemáticos y sus aplicaciones en sistemas dinámicos, sentando así las bases teóricas necesarias para el desarrollo de la implementación.

Se diseñó e implementó una solución computacional que utiliza ecuaciones diferenciales neuronales para simular el crecimiento de tumores. Esta implementación permitió resolver de manera efectiva el modelo matemático que describe la dinámica de crecimiento tumoral sin retardo. Los resultados obtenidos a partir de esta implementación ofrecen una representación detallada y precisa de cómo las células tumorales interactúan y se propagan en un entorno biológico simulado, destacando la utilidad y el potencial de las ecuaciones diferenciales neuronales en la modelización de fenómenos biológicos complejos como el crecimiento tumoral.

Tras la simulación del modelo propuesto, se procedió a evaluar la eficacia de las ecuaciones diferenciales neuronales comparando los resultados simulados con nuevos datos generados y modelos tradicionales de redes neuronales (RNN).

Los resultados de las ecuaciones diferenciales neuronales (NeuralODEs) mostraron un potencial signi-

ficativo en la modelación de sistemas dinámicos complejos, como el crecimiento tumoral. Las NeuralODEs son capaces de capturar las dinámicas fundamentales del sistema mediante la parametrización de un campo vectorial con una red neuronal, permitiendo una representación continua y diferenciable de las trayectorias temporales. Esta capacidad es particularmente útil en la modelación de fenómenos biológicos, donde las variables cambian de manera continua en el tiempo y pueden ser descritas por ecuaciones diferenciales.

En el estudio del crecimiento tumoral, las NeuralODEs demostraron una alta precisión, cuando no se consideran retardos temporales. Las trayectorias predichas por las NeuralODEs se ajustaron bien a los datos observados, reflejando su habilidad para modelar dinámicas complejas que evolucionan de manera continua.

Sin embargo, un desafío significativo surgió cuando se introdujeron retardos temporales en el sistema. Los retardos temporales, comunes en sistemas biológicos debido a procesos de retroalimentación y retrasos en la señalización celular, no fueron manejados de manera efectiva por las NeuralODEs. La precisión de las predicciones disminuyó, indicando que las NeuralODEs, en su forma actual, no están óptimamente equipadas para modelar sistemas con retardos temporales intrínsecos. Este hallazgo sugiere que, aunque las NeuralODEs son herramientas poderosas para ciertas aplicaciones, su uso debe ser evaluado con precaución en escenarios donde los retardos temporales son una característica clave del sistema.

El desempeño de las NeuralODEs se comparó con el de las Redes Neuronales Recurrentes (RNNs), otro tipo de modelo ampliamente utilizado en la modelación de series temporales. Los experimentos mostraron que las NeuralODEs tenían una ventaja significativa en términos de precisión para la predicción de variables dinámicas sin retardo temporal. Esta ventaja se evidenció en métricas de evaluación como el error cuadrático medio (MSE), donde las NeuralODEs lograron un menor error en comparación con las RNNs, indicando una mayor capacidad para capturar las dinámicas subyacentes del sistema.

Sin embargo, al introducir retardos temporales, las diferencias en el rendimiento entre NeuralODEs y RNNs se redujeron considerablemente. Las RNNs, a pesar de ser conceptualmente más simples y menos intensivas computacionalmente, mostraron un rendimiento similar al de las NeuralODEs en la presencia de retardos. Este resultado es significativo, ya que sugiere que para problemas específicos donde los retardos temporales son críticos, la simplicidad y eficiencia computacional de las RNNs pueden hacerlas una opción preferible.

Este desempeño comparativo subraya la importancia de considerar las características específicas del problema al elegir el modelo adecuado. Aunque las NeuralODEs ofrecen ventajas claras en la modelación de sistemas dinámicos continuos sin retardos, su complejidad adicional y los requisitos computacionales pueden no siempre estar justificados cuando los retardos temporales son una consideración importante.

En trabajos posteriores, sería beneficioso implementar ecuaciones diferenciales neuronales con retardo (NeuralDDEs), una extensión de las NeuralODEs que incluye la capacidad de manejar retardos temporales de manera más efectiva. Debido a lo novedoso de esta técnica, la implementación en esta ocasión resultó muy complicada, pero se continúa trabajando en su desarrollo. Las NeuralDDEs prometen abordar las limitaciones encontradas con las NeuralODEs al modelar sistemas con retardos temporales.

Apéndice A

Análisis de Estacionalidad en Sistemas Dinámicos

El análisis de la estabilidad y el comportamiento dinámico de los sistemas es fundamental en diversas áreas de la ciencia y la ingeniería, incluyendo la biología. Este apéndice tiene como objetivo proporcionar una comprensión detallada de varios conceptos clave relacionados con la estacionalidad y la estabilidad en sistemas dinámicos. Se abordarán temas como los puntos de equilibrio, la estabilidad local y global, las funciones de Lyapunov y la bifurcación de Hopf.

A.1. Punto de Equilibrio

Un punto de equilibrio en un sistema dinámico es un estado en el cual el sistema no cambia con el tiempo. Es decir, si el sistema se encuentra en un punto de equilibrio, permanecerá en ese estado a menos que se produzca una perturbación externa. Matemáticamente, un punto de equilibrio (x^*, y^*) para un sistema de ecuaciones diferenciales se define como el punto donde las derivadas con respecto al tiempo son cero:

$$\dot{x} = f(x, y) = 0$$

$$\dot{y} = g(x, y) = 0$$

A.2. Estabilidad Local

La estabilidad local se refiere a cómo se comporta un sistema dinámico en las proximidades de un punto de equilibrio. Si pequeñas perturbaciones alrededor de un punto de equilibrio se desvanecen con el tiempo

y el sistema regresa al punto de equilibrio, este se considera estable. Por el contrario, si las perturbaciones crecen con el tiempo y el sistema se aleja del punto de equilibrio, este se considera inestable [39].

A.2.1. Análisis de la Matriz Jacobiana

Para un sistema con dos ecuaciones y dos incógnitas, la estabilidad local alrededor de un punto de equilibrio se puede analizar utilizando la matriz Jacobiana. La matriz Jacobiana J se calcula evaluando las derivadas parciales de las funciones del sistema en el punto de equilibrio [43]:

$$J = \begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{pmatrix}$$

La estabilidad del sistema depende de los valores propios (autovalores) de esta matriz Jacobiana. En un sistema de dos ecuaciones y dos incógnitas, basta con analizar la traza (la suma de los elementos de la diagonal principal) y el determinante de la matriz Jacobiana para determinar la naturaleza de los autovalores [17]:

- **Traza ($\text{tr}(J)$):** Es la suma de los elementos de la diagonal principal de la matriz Jacobiana.
- **Determinante ($\det(J)$):** Es el producto de los autovalores de la matriz Jacobiana.

El análisis de la traza y el determinante nos permite determinar la estabilidad del sistema:

1. Si $\det(J) > 0$ y $\text{tr}(J) < 0$, el punto de equilibrio es localmente estable (nodos o espirales estables).
2. Si $\det(J) > 0$ y $\text{tr}(J) > 0$, el punto de equilibrio es localmente inestable (nodos o espirales inestables).
3. Si $\det(J) < 0$, el punto de equilibrio es un punto de silla, lo que indica inestabilidad.

A.2.2. Análisis de Estabilidad

Para entender por qué la traza y el determinante de la matriz Jacobiana son suficientes para analizar la estabilidad local de un sistema de dos ecuaciones y dos incógnitas, primero debemos recordar algunos conceptos fundamentales sobre matrices y autovalores.

Dado un sistema de ecuaciones diferenciales de la forma:

$$\begin{cases} \dot{x} = f(x, y) \\ \dot{y} = g(x, y) \end{cases}$$

en un punto de equilibrio (x^*, y^*) , la matriz Jacobiana J es:

$$J = \begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{pmatrix}_{(x^*, y^*)}$$

Los autovalores λ_1 y λ_2 de la matriz Jacobiana determinan la estabilidad local del sistema. Estos autovalores se encuentran resolviendo la ecuación característica:

$$\det(J - \lambda I) = 0$$

Para una matriz 2×2 , la ecuación característica se expande a:

$$\lambda^2 - (\text{tr}(J))\lambda + \det(J) = 0$$

donde $\text{tr}(J)$ es la traza de J y $\det(J)$ es el determinante de J .

A.2.3. Solución de la Ecuación Cuadrática

La solución de esta ecuación cuadrática proporciona los autovalores:

$$\lambda_{1,2} = \frac{\text{tr}(J) \pm \sqrt{(\text{tr}(J))^2 - 4 \det(J)}}{2}$$

Estos autovalores pueden ser reales o complejos dependiendo del discriminante $(\text{tr}(J))^2 - 4 \det(J)$.

Para determinar la estabilidad del sistema, se analiza el signo de los autovalores λ_1 y λ_2 :

1. Ambos autovalores son reales y negativos ($\lambda_1, \lambda_2 < 0$):

- El punto de equilibrio es un nodo estable.
- Condición: $\text{tr}(J) < 0$ y $\det(J) > 0$.

2. Ambos autovalores son reales y positivos ($\lambda_1, \lambda_2 > 0$):

- El punto de equilibrio es un nodo inestable.
- Condición: $\text{tr}(J) > 0$ y $\det(J) > 0$.

3. **Los autovalores son complejos conjugados con parte real negativa ($\text{Re}(\lambda_{1,2}) < 0$):**

- El punto de equilibrio es un espiral estable.
- Condición: $\text{tr}(J) < 0$ y $\det(J) > 0$.

4. **Los autovalores son complejos conjugados con parte real positiva ($\text{Re}(\lambda_{1,2}) > 0$):**

- El punto de equilibrio es un espiral inestable.
- Condición: $\text{tr}(J) > 0$ y $\det(J) > 0$.

5. **Los autovalores son reales y de signos opuestos ($\lambda_1 \cdot \lambda_2 < 0$):**

- El punto de equilibrio es un punto de silla.
- Condición: $\det(J) < 0$.

Por lo tanto, el análisis de la traza y el determinante de la matriz Jacobiana nos proporciona suficiente información para determinar la naturaleza y estabilidad del punto de equilibrio en un sistema de dos ecuaciones y dos incógnitas. Estos dos valores permiten identificar los diferentes tipos de comportamiento dinámico local sin necesidad de calcular explícitamente los autovalores, simplificando así el análisis.

A.3. Estabilidad Global y Funciones de Lyapunov

La estabilidad global de un sistema dinámico se refiere a la estabilidad de todos los puntos de equilibrio en todo el espacio de estados, no solo en una vecindad pequeña alrededor de cada punto de equilibrio. Un método poderoso para estudiar la estabilidad global es el uso de funciones de Lyapunov.

A.3.1. Funciones de Lyapunov

Una función de Lyapunov es una herramienta matemática utilizada para demostrar la estabilidad de un punto de equilibrio en un sistema dinámico. Se define una función escalar $V(x)$ con las siguientes propiedades [21]:

1. $V(x)$ es positiva definida: $V(x) > 0$ para todo $x \neq x^*$ y $V(x^*) = 0$.
2. La derivada de $V(x)$ a lo largo de las trayectorias del sistema es negativa definida:

$$\dot{V}(x) = \frac{dV}{dt} < 0$$

para todo $x \neq x^*$.

Si se puede encontrar una función $V(x)$ que cumpla estas condiciones, entonces el punto de equilibrio x^* es estable en el sentido de Lyapunov. Si además $\dot{V}(x)$ es estrictamente negativa (es decir, $\dot{V}(x) < 0$ para todo $x \neq x^*$), entonces el punto de equilibrio es global asintóticamente estable.

A.4. Bifurcación de Hopf

La bifurcación de Hopf es un tipo de bifurcación en la que un sistema dinámico pasa de tener un punto de equilibrio estable a exhibir un comportamiento oscilatorio periódico. Esta bifurcación ocurre cuando un par de autovalores complejos conjugados de la matriz Jacobiana cruzan el eje imaginario en el plano complejo, cambiando de tener una parte real negativa a tener una parte real positiva [39].

A.4.1. Condiciones para una Bifurcación de Hopf

Para que ocurra una bifurcación de Hopf, se deben cumplir las siguientes condiciones [21]:

1. **Punto de Equilibrio:** Existe un punto de equilibrio x^* para el sistema dinámico.
2. **Autovalores Complejos Conjugados:** En el punto de equilibrio, la matriz Jacobiana J tiene un par de autovalores complejos conjugados $\lambda_{1,2} = \alpha \pm i\beta$.
3. **Parte Real Cero:** En el momento de la bifurcación, la parte real de los autovalores complejos es cero ($\alpha = 0$), y cruza el eje imaginario.
4. **Condiciones Transversales:** La derivada de la parte real de los autovalores con respecto al parámetro de bifurcación es diferente de cero ($\frac{d\alpha}{d\mu} \neq 0$).

Cuando estas condiciones se cumplen, un pequeño cambio en el parámetro de bifurcación μ puede causar que el sistema transicione de un punto de equilibrio estable a un ciclo límite estable o inestable.

Apéndice B

Introducción a las Redes Neuronales Recurrentes y la Celda GRU

La celda GRU (Gated Recurrent Unit) es un componente fundamental dentro de las Redes Neuronales Recurrentes (RNNs), diseñada para abordar el problema de las dependencias a largo plazo y el degradado del gradiente [7].

B.1. Redes Neuronales Recurrentes (RNN)

Las Redes Neuronales Recurrentes (RNN, por sus siglas en inglés) son un tipo de red neuronal diseñada para reconocer patrones en secuencias de datos, como series temporales o lenguaje natural. A diferencia de las redes neuronales tradicionales, que asumen que las entradas son independientes unas de otras, las RNN tienen conexiones recurrentes que permiten la persistencia de información de estados anteriores. Esta capacidad para recordar información pasada las hace especialmente útiles para tareas donde el contexto y el orden de los datos son importantes.

Las RNN procesan una secuencia de datos un paso a la vez, manteniendo un estado oculto que captura información relevante sobre secuencias anteriores. Este estado oculto se actualiza en cada paso de tiempo, y su valor depende tanto de la entrada actual como del estado oculto previo. Sin embargo, las RNN tradicionales enfrentan problemas cuando se trata de secuencias largas debido al fenómeno conocido como "desvanecimiento del gradiente", que dificulta el aprendizaje de dependencias a largo plazo.

B.1.1. La Celda GRU

Para abordar las limitaciones de las RNN tradicionales, se han desarrollado arquitecturas avanzadas como las Celdas de Unidad Recurrente Gated (GRU, por sus siglas en inglés) y las Long Short-Term Memory (LSTM). En este apéndice, nos centraremos en la celda GRU.

La celda GRU es una variante de las RNN que introduce mecanismos de compuertas para controlar el flujo de información a través de la red. Fue propuesta por Kyunghyun Cho y sus colegas en 2014 como una simplificación de la LSTM. La celda GRU utiliza dos compuertas principales: la compuerta de actualización y la compuerta de reinicio.

1. **Compuerta de Actualización:** Controla cuánto del estado anterior debe ser mantenido y cuánto debe ser actualizado con la nueva información. Esto permite que la celda GRU pueda retener información relevante por períodos de tiempo más largos.
2. **Compuerta de Reinicio:** Determina cuánto de la información pasada debe ser olvidada. Al ajustar esta compuerta, la celda GRU puede descartar información irrelevante o menos importante, lo que ayuda a enfocarse en la información más relevante.

La formulación matemática de una celda GRU es la siguiente:

$$\begin{aligned}
 z_t &= \sigma(W_z \cdot [h_{t-1}, x_t]) \\
 r_t &= \sigma(W_r \cdot [h_{t-1}, x_t]) \\
 \tilde{h}_t &= \tanh(W_h \cdot [r_t * h_{t-1}, x_t]) \\
 h_t &= (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t
 \end{aligned}$$

Donde:

- z_t es la compuerta de actualización.
- r_t es la compuerta de reinicio.
- \tilde{h}_t es el nuevo estado propuesto.
- h_t es el nuevo estado oculto.
- σ representa la función sigmoide, que convierte los valores a un rango entre 0 y 1.

- \tanh es la función tangente hiperbólica, que convierte los valores a un rango entre -1 y 1.

La simplicidad de la celda GRU, comparada con la LSTM, la hace menos costosa computacionalmente y más fácil de implementar, mientras sigue proporcionando una solución eficaz a los problemas de desvanecimiento del gradiente y aprendiendo dependencias a largo plazo. Estas características han hecho de la GRU una elección popular en muchas aplicaciones prácticas de redes neuronales recurrentes.

Apéndice C

Implementaciones

A continuación se listan los códigos que desarrollamos para este documento.

- [Simulaciones numéricas para el sistema tumoral.](#)

Simulaciones numéricas que hemos realizado con un módulo en python llamado `jitcdde` diseñado para resolver ecuaciones diferenciales con retraso.

- [Implementación RNN vs NeuralODE para datos sin retardo](#)

Implementación que compara las NeuralODE y RNN para el sistema de ecuaciones diferenciales sin retardo de tiempo junto con el *grid search* realizado para encontrar la arquitectura adecuada para las NeuralODE.

- [Implementación RNN vs NeuralODE para datos con retardo](#)

Implementación que compara las NeuralODE y RNN para el sistema de ecuaciones diferenciales con retardo de tiempo.

Referencias

- [1] *Delay Differential Equations: Recent Advances and New Directions*, Springer US, Boston, MA, 2009 (en).
- [2] Ping Bi and Heying Xiao, *Bifurcations of Tumor-Immune Competition Systems with Delay*, *Abstract and Applied Analysis* **2014** (April 2014), e723159 (en). Publisher: Hindawi.
- [3] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006 (English).
- [4] James R. Brannan and William E. Boyce, *Differential Equations: An Introduction to Modern Methods and Applications*, Hoboken, NJ, 2015 (Inglés).
- [5] J. C. Butcher, *Numerical methods for ordinary differential equations in the 20th century*, *Journal of Computational and Applied Mathematics* **125** (December 2000), no. 1, 1–29.
- [6] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud, *Neural Ordinary Differential Equations*, *Advances in Neural Information Processing Systems*, 2018.
- [7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, *Learning phrase representations using RNN encoder–decoder for statistical machine translation*, *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, October 2014, pp.–1734@.
- [8] Autores de Optax, *Optax documentation*, Read the Docs, 2024. Accedido: 2024-05-25.
- [9] Odo Diekmann, Sjoerd M. Verduyn Lunel, Stephan A. Van Gils, and Hanns-Otto Walther, *Delay Equations* (J. E. Marsden, L. Sirovich, and F. John, eds.), *Applied Mathematical Sciences*, vol. 110, Springer, New York, NY, 1995.
- [10] A. Dubi and N. Gurvitz, *A note on the analysis of systems with time dependent transition rates*, *Annals of Nuclear Energy* **22** (March 1995), no. 3-4, 215–248 (en).
- [11] Gavin P. Dunn, Lloyd J. Old, and Robert D. Schreiber, *The immunobiology of cancer immunosurveillance and immunoediting*, *Immunity* **21** (August 2004), no. 2, 137–148 (eng).
- [12] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh, *Augmented Neural ODEs*, *Advances in Neural Information Processing Systems*, 2019.
- [13] E. Michael Foster, Michael R. Hosking, and Serhan Ziya, *A Spoonful of Math Helps the Medicine Go Down: An Illustration of How Healthcare can Benefit from Mathematical Modeling and Analysis*, *BMC Medical Research Methodology* **10** (June 2010), no. 1, 60.
- [14] Magda Galach, *Dynamics of the tumor-immune system competition - the effect of time delay*, *International Journal of Applied Mathematics and Computer Science* **13** (2003), no. 3, 395–406 (eng).

- [15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, Illustrated edition, The MIT Press, Cambridge, Mass, 2016 (English).
- [16] John Guckenheimer and Philip Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*, Springer Science & Business Media, 2013 (en). Google-Books-ID: XYlpBAAQBAJ.
- [17] Morris W. Hirsch, Stephen Smale, and Robert L. Devaney, *Differential Equations, Dynamical Systems, and an Introduction to Chaos*, Academic Press, 2013 (en). Google-Books-ID: csYhsrOEh_MC.
- [18] Warren M. Hirsch, Herman Hanisch, and Jean-Pierre Gabriel, *Differential equation models of some parasitic infections: Methods for the study of asymptotic behavior*, Communications on Pure and Applied Mathematics **38** (1985), no. 6, 733–753 (en). [_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160380607](https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160380607).
- [19] Pål Johansen, Tazio Storni, Lorna Rettig, Zhiyong Qiu, Ani Der-Sarkissian, Kent A. Smith, Vania Manolova, Karl S. Lang, Gabriela Senti, Beat Müllhaupt, Tilman Gerlach, Roberto F. Speck, Adrian Bot, and Thomas M. Kündig, *Antigen kinetics determines immune reactivity*, Proceedings of the National Academy of Sciences **105** (April 2008), no. 13, 5189–5194. Publisher: Proceedings of the National Academy of Sciences.
- [20] Subhas Khajanchi and Sandip Banerjee, *Stability and bifurcation analysis of delay induced tumor immune interaction model*, Applied Mathematics and Computation **248** (December 2014), 652–671.
- [21] Hassan K Khalil, *Control of nonlinear systems*.
- [22] Patrick Kidger, *On Neural Differential Equations*, arXiv, 2022. arXiv:2202.02435 [cs, math, stat].
- [23] ———, *diffraction documentation*, 2024. Accessed: 2024-05-25.
- [24] ———, *equinox documentation*, 2024. Accessed: 2024-05-25.
- [25] Diederik P. Kingma and Jimmy Ba, *Adam: A Method for Stochastic Optimization*, arXiv, 2017. arXiv:1412.6980 [cs].
- [26] Yang Kuang, *Delay Differential Equation with Application in Population Dynamics* (January 1993).
- [27] Vladimir A. Kuznetsov, Iliya A. Makalkin, Mark A. Taylor, and Alan S. Perelson, *Nonlinear dynamics of immunogenic tumors: Parameter estimation and global bifurcation analysis*, Bulletin of Mathematical Biology **56** (March 1994), no. 2, 295–321 (en).
- [28] Zhilu Lai, Charilaos Mylonas, Satish Nagarajaiah, and Eleni Chatzi, *Structural identification with physics-informed neural ordinary differential equations*, Journal of Sound and Vibration **508** (September 2021), 116196.
- [29] Hans Petter Langtangen and Geir K. Pedersen, *Scaling of Differential Equations*, Springer International Publishing, Cham, 2016 (en).
- [30] Jianquan Li, Xin Xie, Dian Zhang, Jia Li, and Xiaolin Lin, *Qualitative analysis of a simple tumor-immune system with time delay of tumor action*, Discrete and Continuous Dynamical Systems - B **26** (November 2020), no. 10, 5227–5249 (en). Publisher: Discrete and Continuous Dynamical Systems - B.
- [31] J. E. Marsden and M. McCracken, *The Hopf Bifurcation and Its Applications*, Springer Science & Business Media, 2012 (en). Google-Books-ID: FTv0BwAAQBAJ.
- [32] Naoshi Nishida and Masatoshi Kudo, *Immunological Microenvironment of Hepatocellular Carcinoma and Its Clinical Implication*, Oncology **92** (October 2016), no. Suppl. 1, 40–49.
- [33] Kornel Papik, Bela Molnar, Rainer Schaefer, Zalan Dombovari, Zsolt Tulassay, and Janos Feher, *Application of neural networks in medicine-a review*, Med Sci Monit **4** (1998), no. 3, 538–546.

- [34] Jesús Prieto, Ignacio Melero, and Bruno Sangro, *Immunological landscape and immunotherapy of hepatocellular carcinoma*, Nature Reviews Gastroenterology & Hepatology **12** (December 2015), no. 12, 681–700 (en). Publisher: Nature Publishing Group.
- [35] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman, *Universal Differential Equations for Scientific Machine Learning*, arXiv, 2021. arXiv:2001.04385 [cs, math, q-bio, stat].
- [36] W. Sarlet, E. Engels, and L.Y. Bahar, *Time-dependent linear systems derivable from a variational principle*, International Journal of Engineering Science **20** (January 1982), no. 1, 55–66 (en).
- [37] L.F. Shampine and S. Thompson, *Solving DDEs in Matlab*, Applied Numerical Mathematics **37** (June 2001), no. 4, 441–458 (en).
- [38] Hal Smith, *An Introduction to Delay Differential Equations with Applications to the Life Sciences*, Texts in Applied Mathematics, vol. 57, Springer, New York, NY, 2011 (en).
- [39] Steven H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*, 2nd ed., CRC Press, Boca Raton, 2019.
- [40] Gustaf Söderlind, *Automatic Control and Adaptive Time-Stepping*, Numerical Algorithms **31** (December 2002), no. 1, 281–310 (en).
- [41] Horst R. Thieme, *Convergence results and a Poincaré-Bendixson trichotomy for asymptotically autonomous differential equations*, Journal of Mathematical Biology **30** (August 1992), no. 7, 755–763 (en).
- [42] Ch. Tsitouras, *Runge-Kutta pairs of order 5(4) satisfying only the first column simplifying assumption*, Computers & Mathematics with Applications **62** (July 2011), no. 2, 770–775.
- [43] Ferdinand Verhulst, *Nonlinear Differential Equations and Dynamical Systems*, Springer Science & Business Media, 2006 (en). Google-Books-ID: oyaFXgcFJ2cC.
- [44] Hao Xie, Linfeng Zhang, and Lei Wang, *Ab-initio study of interacting fermions at finite temperature with neural canonical transformation*, Journal of Machine Learning **1** (June 2022), no. 1, 38–59. arXiv:2105.08644 [cond-mat, physics:physics].
- [45] Rui Xu and Zhien Ma, *Global stability of a delayed SEIRS epidemic model with saturation incidence rate*, Nonlinear Dynamics **61** (July 2010), no. 1, 229–239 (en).
- [46] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola, *Dive into Deep Learning*, 1st edition, Cambridge University Press, 2023 (English).
- [47] Qunxi Zhu, Yao Guo, and Wei Lin, *Neural Delay Differential Equations*, arXiv, 2021. arXiv:2102.10801 [nlin].
- [48] Qunxi Zhu, Yifei Shen, Dongsheng Li, and Wei Lin, *Neural Piecewise-Constant Delay Differential Equations*, arXiv, 2022. arXiv:2201.00960 [nlin].
- [49] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James S. Duncan, *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients*, arXiv, 2020. arXiv:2010.07468 [cs, stat].
- [50] Dennis G. Zill, *Ecuaciones diferenciales con aplicaciones de modelado*, 8ª ed, Thomson, México, 2007.