UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA QUÍMICA, BIOTECNOLOGÍA Y MATERIALES

APLICACIÓN DE MACHINE LEARNING PARA UN SISTEMA GENERADOR DE MODELOS DIFERENCIALES DE SISTEMAS BIOLÓGICOS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN BIOTECNOLOGÍA E INGENIERO CIVIL QUÍMICO

RAIMUNDO TOMÁS OYARCE ANDRADE

PROFESOR GUÍA:
J. CRISTIAN SALGADO HERRERA

MIEMBROS DE LA COMISIÓN:
ÁLVARO OLIVERA NAPPA
ZIOMARA GERDTZEN HAKIM

SANTIAGO DE CHILE
2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE: INGENIERO CIVIL QUÍMICO
E INGENIERO CIVIL EN BIOTECNOLOGÍA
POR: RAIMUNDO TOMÁS OYARCE ANDRADE
FECHA: 27/09/2021
PROF. GUÍA: J. CRISTIAN SALGADO HERRERA

APLICACIÓN DE MACHINE LEARNING PARA UN SISTEMA GENERADOR DE MODELOS DIFERENCIALES DE SISTEMAS BIOLÓGICOS

En ingeniería de procesos, la formulación de modelos matemáticos es de gran utilidad para diversas aplicaciones, entre las que se encuentran el diseño, el control y/o la optimización de un proceso. La construcción de estos modelos se facilita con Machine Learning debido a su capacidad de identificar fácilmente patrones a partir de los datos, no necesitar intervención humana en el proceso y poder manejar una gran cantidad de datos simultáneamente. Específicamente, los algoritmos de regresión simbólica permiten identificar modelos matemáticos a partir de datos experimentales.

El objetivo de este trabajo es desarrollar y evaluar una metodología que permita proponer funciones o bloques faltantes en modelos matemáticos diferenciales de sistemas biológicos o biotecnológicos mediante regresión simbólica. Esta metodología es evaluada en dos casos de estudio sintéticos: crecimiento de microorganismos en reactor batch y producción simultánea de bioetanol y fructosa a partir de sacarosa. Para la regresión simbólica se utilizó el algoritmo de programación genética (GP).

La metodología diseñada consiste en obtener los datos experimentales, ejecutar el algoritmo GP para identificar la estructura matemática del bloque faltante, ajustar los parámetros de la expresión propuesta por GP y finalmente validar el modelo para medir su capacidad de generalización.

Para la implementación del algoritmo GP se diseñó una función de evaluación que forma un sistema de ecuaciones a partir de las ecuaciones conocidas y la expresión matemática propuesta por cada individuo para el bloque faltante. Dicho sistema es resuelto para luego calcular el fitness del individuo.

La metodología propuesta permite obtener modelos matemáticos con buena capacidad de generalización en ambos casos de estudio, siendo la etapa de validación fundamental para descartar los casos de sobreajuste. Sin embargo, tiene la limitación de necesitar conocer todas las ecuaciones del sistema y su relación con el bloque faltante que se desea identificar. Además, se necesitan dos conjuntos de datos obtenidos en distintas condiciones, uno para el entrenamiento y el otro para la validación.

Agradecimientos

Dejando en claro que una página no basta para plasmar lo agradecido que estoy con un montón de personas por, de una u otra manera me ayudaron a llegar a este momento, quiero agradecer a:

Mi profesor guía, J. Cristian Salgado, por todo su apoyo, comentarios y sugerencias que me permitieron que este trabajo haya salido de la mejor manera posible.

Elizabeth, por siempre preocuparse de mi bienestar tanto físico como mental, por ser mi mayor fuente de motivación para intentar todos los días ser una mejor versión de mi mismo. Te amo y no puedo esperar a seguir viviendo el resto de nuestras vidas juntos.

Mis padres, por hacer posible que llegara hasta aquí, siempre preocupándose de que tuviera todo lo necesario para poder enfocarme completamente en mis estudios.

Mi hermano Felipe, por ayudarme a redactar este informe y por estar continuamente motivándome y dándome ideas sobre mis siguientes pasos en mi vida profesional.

Mi hermano Javier, por compartir varios intereses conmigo y ser, sin darse cuenta, de gran ayuda en los momentos en que necesitaba despejarme de este trabajo.

Mis abuelos: abuela Betty, por acogerme en su hogar y hacer más fáciles mis primero años en la universidad. Abuela María y abuelo Jorge, por tener las puertas de su casa abiertas para mi siempre que lo necesito y por llevarme a la universidad cuando me quedaba a dormir allá.

Gina por toda su orientación y ayuda en este proceso de titulación, que no ha sido para nada fácil.

Todos mis amigos con los que he tenido la suerte de compartir, sea vía mensaje o presencialmente, y brindarme un momento de distracción en este largo proceso de titulación: Dani, Andrea, Nico, Gualla, Feña, Simón, y muy probablemente se me escapa alguno en estos momentos, pido perdón por eso.

Quién se está dando el tiempo de leer este trabajo, del cuál estoy muy orgulloso.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Antecedentes generales

La formulación de modelos matemáticos es una práctica común en ingeniería de procesos. El estudio detallado de un fenómeno científico suele conducir a la proposición de una representación o modelo matemático capaz de simular su comportamiento. En particular, los modelos matemáticos de alta fidelidad permiten un mejor entendimiento de los distintos fenómenos que ocurren dentro del sistema investigado junto con la comprensión de su comportamiento general y la extrapolación a condiciones distintas a las que se tenían durante las etapas de experimentación que permitieron la formulación del modelo [1]. Por lo tanto, en los últimos años ha aumentado el interés por la utilización técnicas de inteligencia computacional para modelar y monitorear diferentes aspectos relacionados con los bioprocesos, algunos ejemplos de ello son: la predicción de la estructura tridimensional de proteínas [2], la predicción de la afinidad entre un ligando y un sitio de unión [3] o la optimización de bioprocesos batch a través de la utilización de aprendizaje reforzado [4],

Los modelos matemáticos pueden servir un propósito técnico, como determinar el balance de masa o de materiales utilizados en una operación unitaria o en el proceso completo, como también pueden tener un propósito de orientar decisiones de negocio [5]. Esto facilita la toma de decisiones técnicas y financieras simultáneamente, permitiendo que una síntesis de procesos óptima más holística se complete en menos tiempo que si se hiciera exclusivamente mediante experimentación. Los modelos ayudan a guiar el ciclo de ensayo y error en la exploración de variables experimentales a utilizar, reduciendo así el tiempo que se pasa en el laboratorio o en la planta piloto [6].

Debido a que gran parte de los fenómenos que ocurren en un bioproceso son de naturaleza transiente, comúnmente su modelamiento se basa en un conjunto de ecuaciones diferenciales que representan balances de masa o de energía. Encontrar dicho conjunto de ecuaciones y sus parámetros para representar apropiadamente al sistema es una tarea que puede tomar mucho tiempo, debido principalmente a la complejidad inherente de dichos procesos, como por ejemplo la no linealidad de las reacciones biológicas. Por lo tanto, en los últimos años ha aumentado el interés por la utilización técnicas de inteligencia computacional para modelar y monitorear diferentes aspectos relacionados con los bioprocesos [7].

Históricamente, la utilización de programas computacionales para el diseño y optimiza-

ción de bioprocesos había sido relativamente poco común en comparación con otras áreas, como la ingeniería química [8]. Sin embargo, más recientemente, los crecientes costos de fabricación y los plazos más cortos disponibles para el desarrollo han aumentado el interés en las simulaciones matemáticas para el diseño de procesos y la evaluación de diagramas de flujo por razones técnicas, de recursos, de planificación y económicas. La necesidad de cambiar los procesos existentes para satisfacer demandas inesperadamente altas del mercado o para reducir el impacto de ineficiencias en las operaciones unitarias que lo conforman generan factores de costo adicionales en el modelamiento y simulación de procesos [9]. Los casos más simples, como por ejemplo un cálculo directo del balance de masa, pueden realizarse de forma personalizada utilizando una hoja de cálculo. Sin embargo, este enfoque se torna más complejo cuando se necesita representar alguna propiedad dinámica del proceso, como por ejemplo la asignación de recursos y su impacto en el rendimiento de la planta [10]. En estas circunstancias, las simulaciones computacionales de modelos matemáticos dinámicos pueden ayudar a mejorar la eficiencia de las actividades de diseño. Estos métodos pueden ser utilizados en una etapa temprana del diseño de un proceso para evaluar múltiples balances de materia y condiciones de operación rápidamente, potencialmente disminuyendo la necesidad de realizar experimentos en plantas pilotos y permitiendo focalizar el trabajo a futuro en las estrategias de producción más factibles. En consecuencia, se pueden tomar decisiones anticipadas y con la debida atención a aspectos como las restricciones de espacio para las instalaciones y/o de índole corporativa que pueden afectar la elección de operaciones unitarias que componen el proceso.

Entonces, el modelamiento matemático puede ser una técnica eficaz para combatir las presiones de costo y tiempo en una amplia variedad de escenarios en los que se requiere la toma de decisiones relacionadas con el proceso productivo. Las simulaciones de los modelos matemáticos reducen los costos y el tiempo necesarios para conducir el desarrollo de procesos y actividades de manufactura maximizando su productividad [11], por lo que son un método útil para diseñar, desarrollar y evaluar bioprocesos [12].

1.2. Motivación

Dada la importancia de la utilización de modelos matemáticos diferenciales en ingeniería de procesos de procesos químicos y biotecnológicos, la motivación del presente trabajo de título es diseñar una metodología que permita utilizar datos experimentales para identificar modelos diferenciales, o parte de ellos, de fenómenos del área de la biotecnología. De ser exitosa, está metodología también podría ser utilizada para proponer modelos matemáticos de otras disciplinas.

1.3. Objetivos

1.3.1. Objetivo general

- Desarrollar y evaluar una metodología que permita proponer funciones o bloques faltantes en modelos matemáticos diferenciales de sistemas biológicos o biotecnológicos.

1.3.2. Objetivos específicos

- Definir casos de estudio, seleccionando modelos matemáticos de problemas representativos de Ingeniería Civil en Biotecnología.
- Evaluar viabilidad de encontrar funciones desconocidas dentro de modelos diferenciales no lineales de sistemas biológicos utilizando el algoritmo de programación genética.
- Diseñar una función de evaluación de modelos diferenciales para el algoritmo de programación genética.
- Estudiar el efecto de distintos parámetros, como el tamaño de población y el número de generaciones, en el desempeño del algoritmo de programación genética.
- Simular la utilización de la metodología propuesta en un caso de estudio representativo de fenómenos más complejos que pueden encontrarse en la industria de procesos.
- Estudiar el efecto de la utilización de métodos de remuestreo en la capacidad de generalización de los modelos matemáticos.

Capítulo 2

Marco Teórico

2.1. Modelamiento matemático

Un modelo matemático es una representación o una interpretación abstracta, mediante el lenguaje matemático, de una realidad física que es susceptible de análisis y cálculo [13]. Las principales ventajas de la utilización de este tipo de lenguaje son que la matemática es un lenguaje preciso, lo que permite formular ideas e identificar supuestos subyacentes; que es un lenguaje conciso, con reglas bien definidas; y que permite la utilización de operaciones computacionales para realizar cálculos numéricos [14]. Los modelos matemáticos se pueden utilizar con distintos propósitos, como comprender un fenómeno a través de una expresión cuantitativa que lo represente, probar los efectos de distintos componentes en el sistema modelado y hacer predicciones sobre patrones de comportamiento.

Dependiendo del nivel de conocimiento previo del fenómeno estudiado que se utiliza para la construcción del modelo matemático, se hace la distinción entre los modelos mecaniscistas y los modelos empíricos. Los modelos mecaniscistas, también conocidos como mecanicistas, se basan en el conocimiento previo sobre el sistema estudiado y en los datos experimentales. Por otro lado, los modelos empíricos establecen una representación de caja negra del proceso a partir de la observación de datos de entrada y salida.

El enfoque más común en la formulación de modelos matemáticos de bioprocesos es el mecaniscista y para construir este tipo de modelos dinámicos se requiere determinar un mecanismo de reacción, seleccionar de una estructura de modelo cinético apropiada y calibrar y optimizar los parámetros de dicha estructura. La calidad del modelo resultante depende de la información que se pueda obtener de los datos experimentales y de la selección de una estructura de modelo apropiada. Por lo tanto, este tipo de modelos pueden ser inexactos, y si bien proporcionan información importante sobre el sistema estudiado, no siempre predicen su comportamiento en un rango amplio de condiciones de operación [15].

Una alternativa para evitar los pasos de identificación requeridos para la construcción de modelos mecaniscistas es el enfoque empírico, que estudia el sistema solo a través de la evolución de las variables de entrada y de salida. Una ventaja significativa de este tipo de modelos es la habilidad de utilizar toda la información contenida en cada medición disponible [15].

2.2. Modelos de sistemas biológicos

Ejemplos de modelos matemáticos utilizados en biotecnología son los de cinética, ya sea enzimática y/o de crecimiento celular, y los modelos metabólicos a escala genómica. Los primeros pueden utilizarse para diseñar, desarrollar y/o evaluar bioprocesos [12]. Por su parte, los modelos metabólicos a escala genómica son útiles para determinar que vías o reacciones limitan el crecimiento del microorganismo o la producción de determinados metabolitos de interés, facilitando el rediseño de medios de cultivo o la aplicación de manipulaciones genéticas con el objetivo de potenciar las características deseadas [16]. A continuación se presentan los modelos de los sistemas biológicos que se estudian en la presente memoria.

2.2.1. Crecimiento de microorganismos en reactor batch

En esta sección se presentan las ecuaciones diferenciales que describen la tasa de crecimiento y la tasa de consumo de sustrato de un microorganismo genérico en un bioreactor batch que opera a temperatura y pH constante, y que cuenta con un único sustrato en el medio. La tasa de crecimiento está representada por la Ecuación 2.1, donde X (g/L) representa la concentración del microorganismo y μ (h^{-1}) su tasa de crecimiento específica.

$$\frac{dX}{dt} = \mu \cdot X \tag{2.1}$$

Por otro lado, la tasa de consumo del sustrato está descrita por la ecuación 2.2, donde S (g/L) representa la concentración del sustrato y $Y_{X/S}$ (g/g) el coeficiente de rendimiento de producción de biomasa a partir del sustrato.

$$\frac{dS}{dt} = -\frac{1}{Y_{X/S}} \cdot \mu \cdot X \tag{2.2}$$

Existen distintos modelos para representar tasa de crecimiento específico, y uno de los más comunes corresponde a la ecuación de Monod. Esta ecuación describe a la tasa de crecimiento específico mediante una curva de saturación que depende de la concentración de sustrato. Este modelo se presenta en la Ecuación 2.3, donde μ_{max} (h^{-1}) corresponde a la tasa de crecimiento específico máxima y K_S (g/L) la constante de saturación, que representa la concentración de sustrato en la que μ es la mitad mitad de μ_{max}.

$$\mu = \mu_{max} \cdot (\frac{S}{K_S + S}) \tag{2.3}$$

Como se mencionó anteriormente, la tasa de crecimiento específico de los microorganismos puede estar descrita por otros modelos, entre los que se encuentran modificaciones a la ecuación de Monod para representar inhibición al crecimiento del microorganismo por una alta concentración de sustrato, como el propuesto por Webb (1963) [17] o el propuesto por Andrews y Noack (1968) [18], o para representar inhibición por un aumento en la concentración de algún producto, como el modelo propuesto por Ghose y Tyagi (1979) [19] o el propuesto por Levenspiel (1980) [20]. Además, μ puede ser modelado por la ecuación logística o ecuación de Verhuslt [21] en función de la capacidad de carga, que corresponde al tamaño máximo de población que el ambiente puede soportar con los recursos disponibles [22].

2.2.2. Producción de bioetanol y fructosa a partir de sacarosa

El etanol es un producto que puede tener varias aplicaciones, ya que es utilizado como solvente, como combustible, como reactivo químico o como materia prima para la producción de varios productos importantes [23]. Un ejemplo reciente de su uso como materia prima es en la producción de alcohol gel, ampliamente utilizado como sanitizante de manos en el marco de la pandemia del Covid-19 [24]. La demanda global de este producto se encuentra en constante crecimiento, especialmente para su utilización como fuente de energía [25], debido a que se proyecta que la producción mundial de combustibles fósiles disminuya desde 25 mil millones de barriles anuales en 2002 a 5 mil millones de barriles en 2050 [26]. El etanol puede ser producido de manera química, mediante la hidratación de etileno, o biológicamente, por la fermentación de azúcares obtenidos de diversas materias primas [27]. El bioetanol corresponde al etanol que es producido de manera biológica, y en comparación con los combustibles fósiles, tiene las ventajas de ser renovable, no tóxico, fácil de manipular, seguro de almacenar y generar una menor contribución al calentamiento global y a la contaminación atmosférica por ser libre de sulfuros [28].

Las materias primas utilizadas para la producción de bioetanol se clasifican en tres tipos: azúcares simples, almidón y celulosa [29]. Los azúcares simples, obtenidos a partir de la caña de azúcar, de la remolacha azucarera, de la melaza y/o de frutas, pueden ser convertidos a etanol directamente por los microorganismos. El almidón, obtenido del maíz, de la yuca, de papas y otros tubérculos, debe ser hidrolizado en azúcares fermentables mediante la acción de enzimas amilasas, presentes en la malta y en hongos filamentosos. La celulosa utilizada proviene principalmente de madera, residuos agrícolas y residuos líquidos de la producción de papel, y también debe ser preprocesada para obtener azúcares simples, generalmente por la acción de ácidos inorgánicos. Una vez obtenidos estos carbohidratos, las enzimas de los microorganismos pueden producir etanol.

La melaza es un producto secundario del procesamiento de azúcar y puede ser utilizado como una fuente de carbono relativamente barata, por lo que es una de las materias primas más comunes para la fermentación de bioetanol [30] [31]. Entre un 30 % y un 50 % de la masa de este subproducto corresponden a sacarosa, que es un disacárido compuesto por glucosa y fructosa [31] [32] [33]. Durante la fermentación con levaduras, estos microorganismos tienen una preferencia por la glucosa por sobre la fructosa, lo que resulta en una diferencia entre el consumo de ambos azúcares a lo largo del proceso [34]. En consecuencia, en las últimas etapas de la fermentación existe una mayor concentración de fructosa, que debe ser metabolizada por microorganismos que probablemente se encuentren condiciones de stress, tales como la falta de nitrógeno o una alta concentración de etanol [35].

Por lo tanto, en la producción de bioetanol a partir de sacarosa es posible obtener fructosa como producto secundario [36]. De hecho, la fermentación selectiva de glucosa y otros azúcares (menos la fructosa) a etanol corresponde a una técnica desarrollada para la obtención simultánea de etanol y fructosa [37]. Además, este proceso permite facilitar la purificación de la fructosa generada en la hidrólisis de sacarosa. La fructosa es el azúcar natural más dulce que existe, por lo que es ampliamente utilizado como endulzante de alimentos y bebestibles [38]. Como glucosa y fructosa son isómeros su separación es compleja, y debe realizarse usando técnicas de cromatografía que pueden llegar a ser muy costosas [39]. Entonces, la conversión

de glucosa en etanol, que es más fácil de separar de la fructosa, es un buena alternativa para obtener un producto con alta concentración de este endulzante [40].

Las levaduras (*Saccharomyces cerevisae*) tienen tres enzimas que catalizan la fosforilación, que es el primer paso en la ruta metabólica de los azúcares: hexoquinasa A (gen HXK1), hexoquinasa (gen HXK2) y una glucoquinasa (gen GLK1) [41] [42]. Las hexoquinasas pueden fosforilar fructosa, glucosa y manosa, mientras que la glucoquinasa sólo fosforila glucosa y manosa [43] [44]. La cepa ATCC 36858 de levaduras tiene mutaciones en los genes HXK1 y HXK2 que causan que no sinteticen hexoquinasas, por lo que no pueden fosforilar fructosa, y en consecuencia, no la metabolizan [45]. Sulieman y col. (2018) plantean un modelo matemático que describe la fermentación selectiva de glucosa en bioetanol a partir de sacarosa utilizando la cepa de levaduras ATCC 36858 en un reactor batch de volumen constante operando a temperatura constante [46]. Este modelo está descrito por las Ecuaciones 2.4-2.10. La Ecuación 2.4 presenta la tasa de producción de biomasa, donde X (g/L) es la concentración de levadura y μ (h^{-1}) es la tasa de específica crecimiento, que considera inhibición por producto (etanol) de acuerdo al modelo de Ghose y Tyagi (Ecuación 2.5) [19].

$$\frac{dX}{dt} = \mu \cdot X \tag{2.4}$$

$$\mu = \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E_m}) \tag{2.5}$$

Donde μ_{max} (h^{-1}) es la tasa de crecimiento específico máxima de la levadura , G (g/L) y E (g/L) son la concentración de glucosa y etanol, respectivamente, K_G (g/L) es la constante de saturación (o constante de velocidad media) para el crecimiento de la levadura y E_m (g/L) es la concentración mínima de etanol a partir de la que se inhibe el crecimiento del microorganismo. Cuando la concentración de etanol en el medio sobrepasa E_m, comienza a producirse la muerte celular. La Ecuación 2.6 describe la tasa de consumo de sacarosa para formar glucosa y fructosa, donde S (g/L) es la concentración de sacarosa, α_s (h^{-1}) es la tasa específica de asimilación de sacarosa por parte de la levadura y K_S (g/L) es la constante de saturación de sacarosa.

$$\frac{dS}{dt} = -\alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \tag{2.6}$$

La Ecuación 2.7 describe la evolución de la concentración de glucosa (G). El primer término de dicha ecuación describe la producción de glucosa a partir de sacarosa, mientras que el segundo bloque corresponde al consumo de glucosa debido al metabolismo de la levadura, produciendo biomasa y etanol. El término $Y_{G/S}$ (g/g) corresponde al coeficiente de rendimiento de producción de glucosa a partir de sacarosa, $Y_{X/G}$ (g/g) corresponde al coeficiente de rendimiento de producción de biomasa a partir de glucosa y q_E (h^{-1}) es la tasa específica de producción de etanol, descrita por la Ecuación 2.8.

$$\frac{dG}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{G/S} - q_E \cdot X \cdot \frac{1}{Y_{X/G}} \tag{2.7}$$

$$q_E = \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E'_m}) \tag{2.8}$$

La tasa de producción de etanol también está descrita por el modelo de Ghose y Tyagi, dando cuenta que la síntesis de etanol es inhibida por este producto. E'_m (g/L) corresponde a la concentración umbral de etanol a partir de la cual las células dejan de producirlo. Cabe destacar que es común que la constante E'_m sea mayor que E_m, lo que indica que el crecimiento del microorganismo fuertemente inhibido por la presencia de etanol [47]. La Ecuación 2.9 describe la evolución de la concentración de fructosa. La primera parte de esta ecuación corresponde a la formación de fructosa a partir de la hidrólisis de sacarosa y la segunda parte describe el consumo de fructosa en la producción etanol, el que está ligado al consumo de glucosa, por lo que depende de q_E.

$$\frac{dF}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{F/S} - q_E \cdot X \cdot \frac{R}{Y_{X/G}} \tag{2.9}$$

Donde F (g/L) es la concentración de fructosa, $Y_{F/S}$ (g/g) corresponde al coeficiente de rendimiento producción de fructosa a partir de sacarosa y R (g/g) es la proporción entre la tasa de consumo de fructosa y la tasa de consumo de glucosa para la síntesis de etanol. Nótese que $Y_{F/S}$ es equivalente a $Y_{G/S}$, debido a que tanto glucosa como fructosa se obtienen simultáneamente al hidrolizar la sacarosa y que ambos monosacáridos son isómeros, por lo que tienen la misma masa molar. Finalmente, la Ecuación 2.10 describe la evolución de la concentración de etanol, donde $Y_{E/G}$ corresponde al coeficiente de rendimiento de producción de etanol a partir de glucosa (g/g).

$$\frac{dE}{dt} = q_E \cdot X \cdot \frac{Y_{E/G}}{Y_{X/G}} \cdot (R + 1) \tag{2.10}$$

2.3. Machine learning

Machine learning es un subcampo de las ciencias de la computación que se ocupa de construir algoritmos que, para ser utilizados, se basan en una colección de ejemplos de algún fenómenos. Estos ejemplos pueden provenir de la naturaleza, ser creados por el humano o ser generados por otro algoritmo. También puede definirse como el proceso resolución de un problema práctico mediante la recopilación de un conjunto de datos y la construcción algorítmica de un modelo estadístico basado en dicho conjunto de datos. Luego, el modelo estadístico es utilizado de alguna manera para resolver el problema práctico mencionado [48].

Esta forma de Inteligencia Artificial utiliza una variedad de algoritmos que aprenden iterativamente a partir de los datos entregados para mejorar, describir dichos datos y predecir resultados. Por lo tanto, a medida que los algoritmos reciben datos, es posible producir modelos más precisos basados en estos. Sus ventajas son la capacidad de identificar fácilmente patrones a partir de los datos, no necesitar intervención humana en el proceso, poder manejar una gran cantidad de datos simultáneamente y tener un amplio espectro de aplicaciones. Por otro lado, sus principales desventajas son el tiempo que tardan los algoritmos en encontrar soluciones con niveles considerables de exactitud y relevancia, y requerir de altos recursos

computacionales para funcionar [49].

Dependiendo del tipo de problemas que son capaces de resolver, los algoritmos de machine learning pueden clasificar en aprendizaje supervisado y aprendizaje no supervisado [50]. El aprendizaje supervisado corresponde a una técnica para generar una función a partir de datos de entrenamiento, que consisten en variables de entradas (*inputs*) y variables de salida (*outputs*). La misión del aprendizaje supervisado es predecir la variable de salida de la función para cualquier *input* válido luego de haber recibido ejemplos de entrenamiento [51]. Las variables de salida pueden ser cuantitativas o cualitativas. Cuando los *outputs* son cuantitativos se dice que el problema corresponde a una regresión, y cuando son cualitativos se dice que es una clasificación [52].

Las técnicas de aprendizaje no supervisado sólo reciben datos de entrada, sin tener información sobre las variables de salida que estos generan. El objetivo de este tipo de aprendizaje es encontrar patrones en los datos más allá de lo que podría considerarse como ruido no estructurado. Por lo tanto, la finalidad del aprendizaje no supervisado puede ser la construcción de un modelo probabilístico de los datos proporcionados. Los ejemplos típicos son la reducción de dimensión y el agrupamiento (*clustering*) [53].

2.3.1. Regresión

La regresión, también conocida como aproximación de funciones, es una técnica de aprendizaje supervisado que permite comprender la correlación entre variables. Su objetivo es encontrar una función cuya variable de salida cumpla con alguna condición deseada que, en la mayor parte de los casos, corresponde a que sus valores se ajusten a un conjunto de datos experimentales [50]. Para hacerlo, se minimiza la diferencia entre el valor de la variable dependiente estimado por la función propuesta $Y(X)$ y su valor observado $\hat{Y}(X)$, denominada error de predicción o residuo (e). En la Ecuación 2.11 se muestra el residuo de la i-ésimo dato medido [54].

$$e_i = Y_i(X) - \hat{Y}_i(X), \qquad i = 1, ..., n. \tag{2.11}$$

Por lo general, el objetivo de la regresión consiste en encontrar función de manera que la suma de los cuadrados de los residuos (en inglés *sum of squared residuals*, o SSR), enunciada en la Ecuación 2.12, sea mínima [54].

$$SSR = \sum_{i=1}^{n} {e_i}^2 = \sum_{i=1}^{n} \left(Y_i(X) - \hat{Y}_i(X)\right)^2 \tag{2.12}$$

2.3.1.1. Regresión paramétrica

En este tipo de regresión la estructura de la función es conocida y sólo se deben estimar un número determinado de parámetros, es decir, la función predefinida es ajustada a los datos experimentales. La elección del tipo de función se realiza a partir de las características (o forma) de los datos experimentales y ayuda simplificar el proceso de aproximación. Por lo

tanto, la elección del tipo de función corresponde a una adivinanza y realizar suposiciones posiblemente erradas puede resultar en modelos de mala calidad [55].

La regresión lineal corresponde al ejemplo de regresión paramétrica más común y consiste en ajustar los datos experimentales a la ecuación de una recta (Ecuación 2.13) optimizando dos parámetros: la pendiente (p_1) y el coeficiente de posición (p_2)

$$Y(X) = p_1 \cdot X + p_2 \tag{2.13}$$

La desventaja de la regresión de parámetros es que, si el ajuste a una función determinada no es bueno, se debe intentar con diferentes funciones hasta que se logra encontrar un modelo que represente a los datos. Este proceso no sólo puede requerir mucho trabajo, sino que también depende del ingenio, conocimiento y/o experiencia del usuario que propone las funciones que se ajustan. Es probable que se den casos en los que la funcionalidad de los fenómenos es compleja, por lo cual no es posible establecer una función para ajustar [56]. Para estos casos, existe la regresión simbólica, que intenta ir más allá del problema de ajustar parámetros al encontrar una función que se ajuste a los datos disponibles sin realizar ningún supuesto sobre la estructura de dicha función, o bien, estableciendo supuestos generales [57].

2.3.1.2. Regresión simbólica

A diferencia de los métodos de regresión paramétrica, que optimizan los parámetros de una ecuación de estructura conocida, las regresiones simbólicas buscan simultáneamente la forma de la ecuación y sus parámetros. Para lograrlo, las expresiones matemáticas son construidas como combinaciones aleatorias de distintos bloques de construcción, entre los que se pueden encontrar operaciones algebraicas (suma, resta, multiplicación, división), funciones (funciones trigonométricas, exponencial, logarítmo, entre otras), constantes numéricas y variables. Nuevas ecuaciones son formadas mediante la recombinación de las ecuaciones previas y variando sus subexpresiones probabilísticamente. Luego, el algoritmo conserva las ecuaciones que mejor modelan los datos experimentales, eliminando soluciones poco prometedoras. Finalmente, cuando la ecuación alcanza el nivel deseado de exactitud, el algoritmo termina y entrega un conjunto de ecuaciones que mejor corresponden al mecanismo intrínseco subyacente del sistema observado [58].

Existen diversos tipos de algoritmos utilizados para resolver problemas de regresión simbólica, entre los que se encuentran: los algoritmos de programación genética [59], algoritmos de colonia de abejas artificiales [60], algoritmos de hormigas dinámicas [61], algoritmos basados en biogeografía [62] y *fast function extraction* (FFX) [63].

2.3.2. Algoritmos evolutivos

Los algoritmos evolutivos (*evolutionary algorithms*, EA) comprenden una clase de técnicas de optimización que imitan los principios de la evolución biológica. El fundamento principal de los algoritmos evolutivos es aplicar los fenómenos evolutivos, tales como mutación y selección del más apto, a poblaciones de soluciones candidatas con la finalidad de

encontrar una solución lo suficientemente buena para un problema de optimización dado [64].

Otro tipo de algoritmos que pueden clasificarse dentro de los EA son los algoritmos inspirados en la naturaleza. Estos intentan emular el comportamiento de algunos organismos determinados para conducir el proceso de búsqueda de soluciones, ejemplos de estos tipos de algoritmos son los de colonias de hormigas o colonias de abejas. Entre los tipos de EA se encuentran los algoritmos genéticos (*genetic algorithm*, GA), la optimización por enjambre de partículas (*particle swarm optimization*, PSO) y la evolución diferencial (*differential evolution*, DE) [65].

2.3.3. Algoritmos genéticos

Los algoritmos genéticos (GA) son algoritmos evolutivos basados en la reproducción sexual y en el principio darwiniano de la supervivencia del más apto. Los algoritmos genéticos se diferencian del resto de métodos de optimización en cuatro aspectos fundamentales [66]:

1. Trabajan con una codificación del conjunto de parámetros que se buscan optimizar en lugar de hacerlo con los parámetros mismos.
2. Buscan un conjunto de soluciones en lugar de una única solución.
3. No requieren información de gradiente ni conocimiento extra sobre el problema.
4. Otros métodos utilizan reglas de transición deterministas, mientras que los GA utilizan reglas probabilísticas.

Por lo tanto, las principales ventajas de los algoritmos genéticos por sobre otros algoritmos de optimización son la no dependencia de un gradiente, su alta capacidad exploratoria y el paralelismo [67]:

- **No dependencia de gradiente:** No se requiere información de gradiente, sólo la evaluación de la función objetivo. Esto hace que GA sea un método versátil y le entrega la habilidad de resolver problemas que tengan una función objetivo compleja cuya derivada sea difícil de obtener o no pueda ser obtenida (función no diferenciables), como es el caso de las funciones discontinuas o discretas [68].
- **Capacidad exploratoria:** Debido a que los operadores genéticos generan nuevas soluciones de manera estocástica, el espacio de búsqueda se explora en varias direcciones simultáneamente, lo que evita el estancamiento en óptimos locales y hace más probable la obtención del óptimo global.
- **Paralelismo:** Debido a que los individuos de una población actúan como agentes independientes, es posible explorar el espacio de búsqueda en varias direcciones simultáneamente. Esta característica permite la paralelización del algoritmo, ya que se pueden manipular y/o evaluar distintos parámetros e incluso distintos grupos de individuos al mismo tiempo [69].

Para alcanzar la solución a un problema, se inicia con un conjunto inicial de individuos generados aleatoriamente llamado población. Cada uno de estos individuos se denomina cromosoma y representa una posible solución al problema [70]. Como se muestra en la Figura 2.1, un cromosoma es una estructura lineal formada por una secuencia de símbolos que evoluciona a lo largo de una sucesión de iteraciones, denominadas generaciones. Durante cada generación, los cromosomas son evaluados en términos de qué tan bien se desempeñan en el entorno del problema particular. Esta medida se denomina medida de aptitud (en inglés *fitness*) y su naturaleza varía con el problema que se quiere resolver. En el caso de los problemas de regresión, el fitness corresponde a alguna medida de error entre el resultado generado y los datos proporcionados.

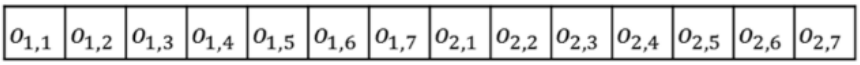


Figura 2.1: Ejemplo de representación en forma de cromosoma de un individuo en GA [71].

Para producir una nueva generación, nuevos cromosomas son creados a partir de los de los individuos de la generación actual utilizando operadores genéticos (entrecruzamiento y mutación). Se dice que los nuevos cromosomas son la descendencia de los cromosomas preexistentes, denominados padres. La nueva generación se forma seleccionando algunos padres de acuerdo a su aptitud, los cromosomas con mejor aptitud tienen mayor probabilidad de ser seleccionados como padres. Luego, se les aplican operadores genéticos para crear su descendencia [72]. Luego de varias generaciones el algoritmo converge al mejor cromosoma, que representa a la solución óptima del problema, o una aproximación a ella. En general, los algoritmos genéticos tienen cinco componentes básicos [73]:

1. Una representación genética de las potenciales soluciones al problema.
2. Una manera de crear una población inicial de estas potenciales soluciones.
3. Una función que evalúe las soluciones de acuerdo a su capacidad de resolución del problema, denominada función de evaluación (*fitness function*).
4. Operadores genéticos que alteren la composición genética de la descendencia.
5. Paramétros propios de los algoritmos genéticos, como el número de individuos por generación, número máximo de generaciones, probabilidades de aplicar cada operador genético, entre otros.

Algunas desventajas de los algoritmos genéticos son que la elección de algunos de sus parámetros, como el tamaño de población, la probabilidad de ocurrencia de los operadores genéticos y la función de evaluación utilizada debe realizarse cuidadosamente, debido a que una selección inapropiada puede generar dificultades en la convergencia del algoritmo. A pesar de estos inconvenientes, los algoritmos genéticos son unos de los métodos más utilizados en la resolución de problemas de optimización no lineal [74].

Otra desventaja de GA es que la representación de las posibles soluciones en estructuras lineales de caracteres de longitud fija puede limitar severamente su capacidad para resolver

algunos problemas, entre ellos los de regresión simbólica, debido a que es preferible que las soluciones tengan distintos tamaños y formas. En estos casos, la representación más útil para una solución es una composición jerárquica de funciones y terminales, denominada programa computacional, utilizados en algoritmos de programación genética [59].

2.3.4. Programación genética

La programación genética (*genetic programming*, GP) es una técnica de computación evolutiva que deriva de los algoritmos genéticos tradicionales, manteniendo el principio de selección natural. La potencialidad de GP reside en que la resolución de problemas se realiza mediante la inducción de programas computacionales, entendiéndose éstos como algoritmos en los que se generan variables de salida a partir de una serie de variables entradas. Por lo tanto, resuelve automáticamente los problemas sin requerir que el usuario conozca o especifique de antemano la forma o estructura de la solución, por lo que es de gran utilidad para deducir ecuaciones matemáticas a partir de datos experimentales [57].

Como se mencionó, la base biológica de GP es exactamente la misma que la de los GA, por lo que su funcionamiento es similar. La diferencia entre una técnica y otra consiste en la forma de codificación de problemas, mientras en GA se realiza en forma de cromosoma en GP se hace en forma de programas computacionales representados como árboles sintácticos binarios, lo que permite su utilización en una serie de entornos donde anteriormente los GA son muy complicados de aplicar, como la regresión simbólica [75].

El algoritmo de programación genética se ha sido utilizado en diversas áreas. Entre sus aplicaciones se encuentran: identificar y corregir automáticamente errores de programación [76], la síntesis automática de circuitos eléctricos analógicos [77], la clasificación de señales de electroencefalogramas [78] y detección de modificaciones no autorizadas a páginas web (*web defacement*) [79]. En la industria de procesos, se usó para estimar la composición del producto de cola en una columna de destilación binaria y la concentración final de un producto en un sistema de dos reactores continuos [80]. Además, se ha utilizado para modelar distintos fenómenos biológicos: la absorción intestinal de hierro [81], fermentación batch de ácido glucónico a partir de glucosa [82], la velocidad de reacción de las enzimas adenilato quinasa y fosfofructoquinasa [83] y para diseñar un sistema de control en la fermentación fed-batch a partir de aceites vegetales [84], entre otros.

2.3.4.1. Representación en árbol

Un árbol es una colección de vértices y aristas. Un vértice, también denominado nodo, es un objeto simple que puede tener un nombre y puede tener información asociada, y una arista es una conexión entre dos vértices. Un camino corresponde a una lista de nodos distintos y sucesivos que están conectados por aristas en el árbol. La propiedad que define a un árbol es que siempre existe un único camino que conecta a dos nodos cualquiera [85]. El tamaño de un árbol corresponde al número de nodos que contiene. En la Figura 2.2 se ilustra un árbol de tamaño 5, en el que se puede evidenciar que existe un único camino entre dos de ellos. Por ejemplo, el camino A-C-D-E es la única manera de conectar el nodo A con el E.

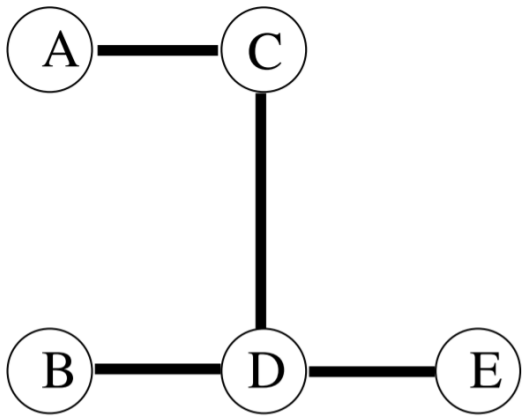


Figura 2.2: Ejemplo de árbol [86].

Por lo general, en ciencias de la computación, el término árbol se refiere a un árbol con raíz, mientras que se utiliza el concepto de árbol libre para referirse a la estructura general ilustrada en la Figura 2.2. Un árbol con raíz es aquel en el que se designa un nodo como la raíz del árbol, y usualmente se dibujan con este nodo en la parte superior. Cada nodo, exceptuando la raíz, tiene exactamente un nodo sobre él, el que se dice que es padre del anterior. De la misma manera, se denominan hijos de un nodo a los nodos que se encuentran directamente bajo él. Lo nodos que no tienen hijos se llaman nodos terminales u hojas del árbol.

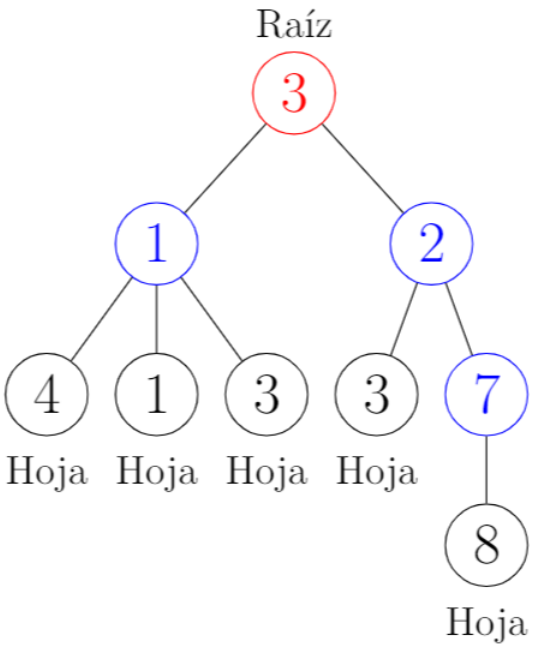


Figura 2.3: Ejemplo de árbol con raíz.

La raíz de un árbol se puede identificar como el único nodo que no tiene padre. Además, en este tipo de árboles cualquier nodo es la raíz de un subárbol, consistente en dicho nodo y todos los nodos que se encuentran bajo él. En la Figura 2.3, se muestra un ejemplo de árbol con raíz, en el que el nodo de color rojo representa a la raíz del árbol completo, mientras que

los nodos azules corresponden a las raíces de los subárboles y los nodos negros corresponden a las hojas del árbol.

En este tipo de árboles, un camino se define como cualquier secuencia de nodos que cumple con que cada nodo es padre del siguiente en la secuencia. La longitud del camino se define como el número de aristas comprendidos entre el nodo de inicio y el nodo final. A partir de la longitud de un camino se definen los conceptos de altura de un nodo, altura del árbol, profundidad de un nodo y nivel de un nodo:

- **Altura de un nodo:** Es la longitud del camino más largo que comienza en el nodo y termina en una hoja. La altura de un nodo hoja es cero, y la altura de cualquier otro nodo se puede calcular sumando uno a la mayor altura de sus hijos.
- **Altura del árbol:** Corresponde a la altura de la raíz. La raíz del árbol que se muestra en la Figura 2.3 tiene altura 3, por lo que el árbol tiene altura 3.
- **Profundidad de un nodo:** Se define como la longitud del camino desde la raíz hasta dicho nodo. La profundidad de la raíz es cero y la profundidad de cualquier otro nodo se puede obtener sumando uno a la profundidad de su padre.
- **Nivel de un nodo:** Corresponde al número de nodos que se encuentran en el camino entre la raíz y el nodo. La raíz se encuentra en el nivel uno, y el nivel de los demás nodos se puede calcular sumando uno al nivel de su padre.

En GP, los individuos de la población son composiciones jerárquicas de funciones primitivas (conjunto F) y terminales (conjunto T) apropiadas para el dominio del problema particular. El conjunto de F típicamente incluye operaciones aritméticas, funciones matemáticas, operaciones lógicas condicionales y funciones específicas del área del problema. El conjunto T generalmente incluye variables y constantes numéricas, pero también puede incluir funciones de aridad cero, es decir, que no necesitan argumentos [87]. Un árbol sintáctico es una estructura que desarrolla la interpretación de un programa computacional. Las funciones se representan por nodos y sus argumentos como hojas de dichos nodos. Un subárbol es una parte de un árbol que se encuentra bajo algún nodo interno [88].

Los árboles sintácticos utilizados para representar a los individuos en GP son binarios, es decir, cada nodo tiene a lo más dos hijos. Estos árboles son recorridos en in-orden. Recorrer un árbol significa visitar cada uno de sus nodos en un orden específico, y visitar un nodo es realizar alguna acción con él, ya sea revisar su información, actualizarla, mostrarla, entre otros. El recorrido in-orden sigue el algoritmo descrito a continuación, comenzando por el nodo raíz [89]:

1. Recorrer el subárbol izquierdo recursivamente de manera in-orden.
2. Visitar nodo actual.
3. Recorrer el subárbol derecho recursivamente de manera in-orden.

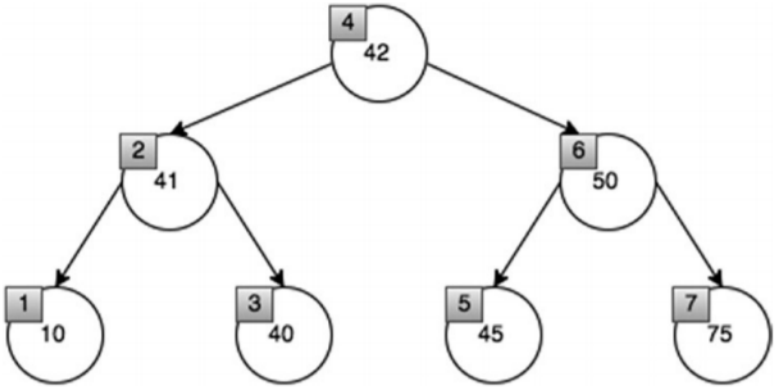


Figura 2.4: Recorrido in-orden de árbol binario [90].

En la Figura 2.4 se ilustra un ejemplo de recorrido in-orden. En este caso, se recorre recursivamente hacia la izquierda hasta llegar a un nodo hoja, que es el nodo del que se extrae información primero (10), luego se lee su nodo padre (41) y se continúa recorriendo el árbol recursivamente hacia la derecha (40). El nodo que se encuentra más a la izquierda del árbol es el primero en ser leído, mientras que el que se encuentra más a la derecha es el último. El orden de recorrido es: 10-41-40-42-45-50-75.

De esta manera, la representación en forma de árbol sintáctico binario está formada por dos tipos de nodos: funciones y terminales. Los nodos de funciones son los que tienen uno o más nodos hijos, y contienen elementos del conjunto F. Por su parte, los nodos terminales, también denominados hojas del árbol, son aquellos que no tienen hijos y contienen elementos del conjunto T. En la Figura 2.5 se muestra un ejemplo de árbol, que representa la expresión matemática $x_3 \cdot (x_2 - 3) + x_1$. En este caso, las funciones corresponden a la adición, la multiplicación y la sustracción, mientras que los nodos terminales son las variables x_1, x_2 y x_3, junto con la constante 3.

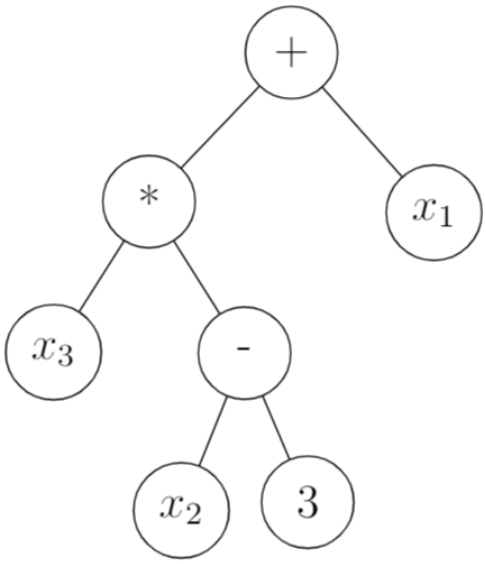


Figura 2.5: Representación en árbol de la expresión $x_3 \cdot (x_2 - 3) + x_1$.

Cabe destacar que una expresión matemática puede ser representada por más de un árbol. En la Figura 2.6 se muestra un ejemplo de la representación en árbol de dos expresiones equivalentes, que también son equivalentes a la mostrada en la Figura 2.5.

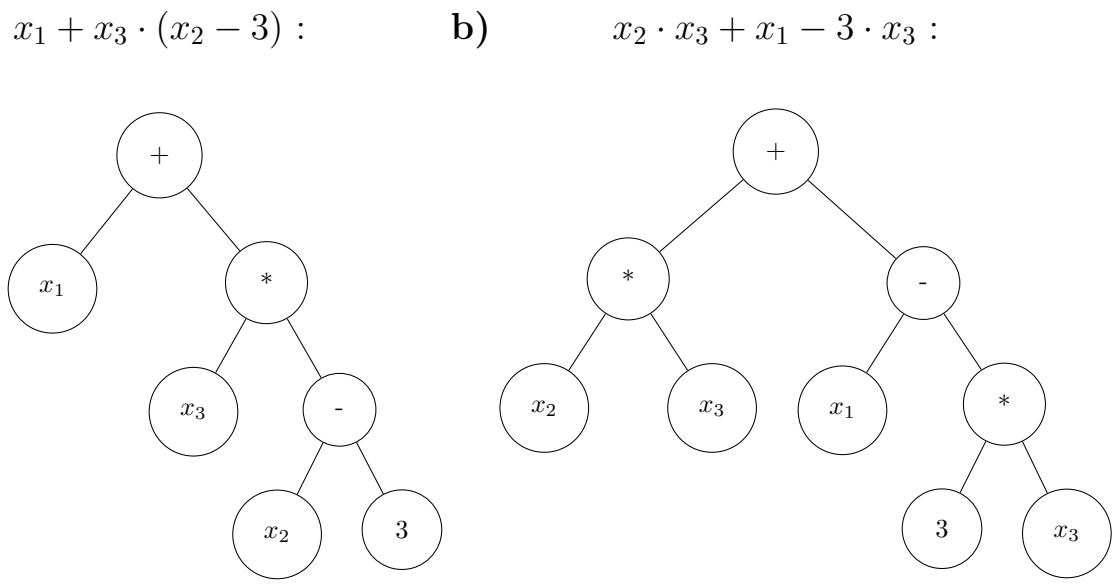


Figura 2.6: Representación en árbol de dos expresiones matemáticas equivalentes.

2.3.4.2. Requisitos de funciones y terminales

Una parte fundamental del funcionamiento de GP es la especificación de los conjuntos F y T antes del inicio del proceso evolutivo. Por lo tanto, es necesario un mínimo proceso de análisis del problema para configurar el algoritmo, para tener una noción sobre que operadores, funciones, variables y constantes numéricas incluir en cada conjunto. Como regla general, es conveniente utilizar sólo los operadores que sean necesarios, puesto que la inclusión de más elementos provocará que el algoritmo tarde más tiempo en encontrar la solución al problema. Además, al definir los conjuntos F y T se deben cumplir con los requisitos de clausura y suficiencia [87]:

- **Clausura:** Esta propiedad requiere que las funciones del conjunto F retornen valores del mismo tipo que sus argumentos. Para que esta condición se cumpla es necesario que cada una de las funciones en el conjunto F sea capaz de utilizar a cualquier constante numérica o variable del conjunto T como su argumento.

  Existen funciones que tienen su dominio acotado, por ejemplo, la función logaritmo sólo está definida para los números reales positivos. Entonces, es probable que GP cause un error al intentar evaluar alguna de estas funciones en un punto en el que no está definida, por lo que no se podría completar la ejecución del algoritmo. Por lo tanto, para cumplir requisito de clausura se utilizan versiones protegidas de funciones que no estén definidas en algún punto o intervalo [91]. Proteger una función significa ampliar

su dominio a todos los números reales. A continuación se muestra un ejemplo de la función logaritmo protegido (log^*), en la que se define que el logaritmo de un número negativo retorne una constante, que típicamente es cero [64].

$$log^*(x) = \begin{cases} log(x) & si \quad x > 0 \\ \\ 0 & si \quad x \leq 0 \end{cases}$$

- **Suficiencia:** Esta propiedad requiere que el conjunto de funciones y el conjunto de terminales sean capaces de expresar una solución al problema planteado, es decir, que entre todas las expresiones posibles que se pueden generar a partir de los elementos de estos conjuntos exista por lo menos una de ellas que resuelva el problema. Al momento de especificar los conjuntos T y F el usuario debe saber, o por lo menos intuir, que alguna composición entre los elementos de ambos conjuntos puede generar una solución al problema. Como antecedente de funciones seleccionadas en sistemas biológicos, Sugimoto y col. (2005) utilizaron las funciones suma, resta, multiplicación y división para modelar la tasa de reacción enzimática [83].

2.3.4.3. Funcionamiento del algoritmo

Al igual que en GA, cada individuo de la población se mide en términos de qué tan bien se desempeña en el entorno del problema particular a través de la medida de aptitud o fitness. Por lo general, gran parte de los individuos creados en la primera generación (generación 0) no representarán una buena solución al problema planteado. Sin embargo, algunos de ellos tendrán un mejor fitness que el resto. Estas diferencias en el rendimiento son explotadas por el principio darwiniano de reproducción y supervivencia del más apto y por los operadores genéticos de la recombinación sexual (*crossover*) y de mutación para crear una nueva población partir de los individuos de la generación actual. A grandes rasgos, como se ilustra en la Figura 2.7, el algoritmo de programación genética genera expresiones matemáticas para resolver problemas ejecutando los siguientes pasos:

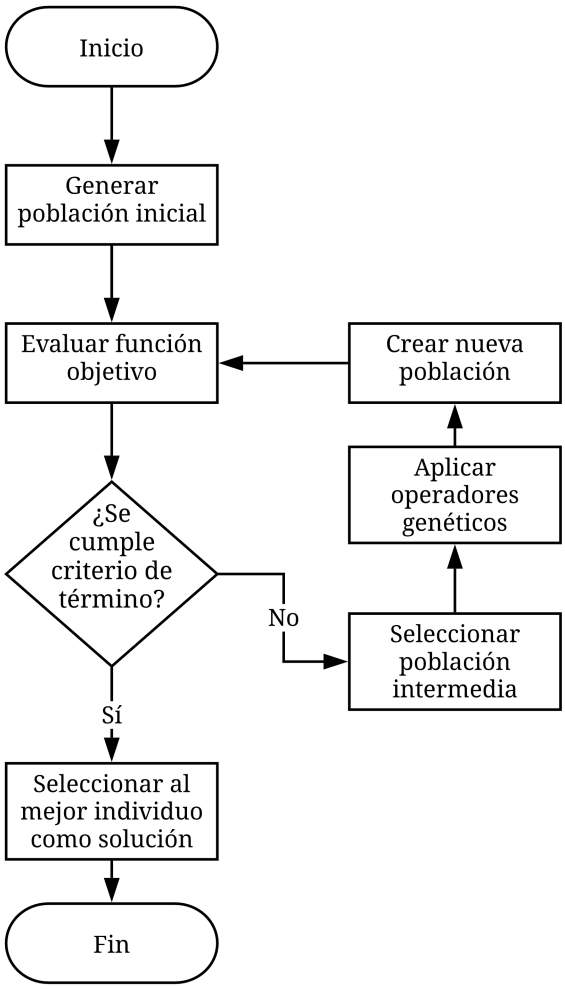
1. Genera una población inicial de expresiones matemáticas aleatorias, denominadas individuos. Estas expresiones están compuestas por funciones (operadores matemáticos, funciones trigonométricas, entre otras) y terminales (variables y/o constantes numéricas).

2. Iterativamente realiza los siguientes pasos hasta que se cumpla el o los criterios de término del algoritmo:

   a) Utiliza a cada individuo para resolver el problema dado, de manera de poder calcular su fitness.

   b) Crea una nueva población al seleccionar individuos mediante probabilidad condicionada por su fitness (mientras mejor fitness mayor probabilidad de que sea seleccionado) y aplicarles operaciones genéticas (entrecruzamiento o mutación).

3. El mejor individuo generado durante la ejecución del algoritmo es designado como el resultado. Dicho resultado puede ser una solución al problema planteado o una aproximación de ella.



.

Figura 2.7: Esquema del algoritmo de programación genética [92].

2.3.4.4. Población inicial

La creación de la población inicial, o inicialización, es la primera fase de GP y es cuando se genera una población de soluciones factibles de manera aleatoria o heurística. A pesar de que esta etapa sólo se ejecuta una vez en el algoritmo, cumple con un papel importante para mejorar su desempeño [93], los individuos de cada generación dependen de la generación anterior hasta llegar a la población inicial [94]. Las probabilidades de obtener la solución óptima aumenta cuando la población inicial es diversa y tiene individuos de buena calidad (buen fitness) [95]. A continuación, se describen ventajas y desventajas de las dos formas principales de generar la población inicial [66].

- **Inicialización aleatoria:** Esta técnica de inicialización genera una población de bajo fitness, por lo que se requiere más tiempo de ejecución para llegar a la solución óptima. Sin embargo, es considerada como la manera más apropiada de crear los individuos iniciales, debido a que se forma una población más diversa que con el método mecaniscista, lo que aumenta la probabilidad de encontrar el óptimo global. Es especialmente utilizada cuando no se tiene conocimiento previo sobre el problema.

- **Inicialización heurística:** Este tipo de inicialización depende de la información que se tenga disponible sobre el problema que se quiere resolver [96]. Si se cuenta con este tipo de información, se puede acotar el espacio de búsqueda, al reconocer áreas en que las soluciones son de alta calidad. Por lo tanto, se genera una población inicial de alto fitness, lo que ha sido probado que permite que se encuentren soluciones cercanas al óptimo más rápidamente [97]. Sin embargo, se puede terminar con espacios de búsqueda muy acotados, lo que puede causar que el algoritmo se estanque en óptimos locales en lugar de encontrar el óptimo global debido a la falta de diversidad en la población [98].

Por lo general, en GP se utilizan métodos de inicialización aleatoria. Hay varias maneras distintas de generar la población inicial, pero los más comunes son *full* y *grow*. En ambos métodos los individuos iniciales se crean sujeto a una restricción en la altura máxima de árbol. También existe una combinación entre *full* y *grow* , denominada *ramped half-and-half*, la cuál es ampliamente utilizada [91]. A continuación, se describen cada una de las técnicas mencionadas anteriormente:

- ***Full*:** Se escogen aleatoriamente nodos desde el conjunto de funciones hasta que se alcanza la altura máxima de árbol establecida y, a partir de ese momento, sólo se pueden escoger nodos desde el conjunto terminal, para no sobrepasar dicha altura. Al forzar la utilización de elementos del conjunto F hasta alcanzar altura de árbol predeterminada, este método se asegura de crear árboles completos.

  Aunque este método genera árboles en que todas las hojas se encuentran en el mismo nivel, esto no necesariamente significa que todos los árboles iniciales tengan el mismo tamaño (número de nodos) o la misma forma. Lo anterior sólo ocurre cuando todas las funciones del conjunto T tienen la misma aridad (cantidad de argumentos). Aún así, incluso cuando se utilizan funciones de distinta aridad el rango tamaños y formas de los árboles creados es limitado [57].

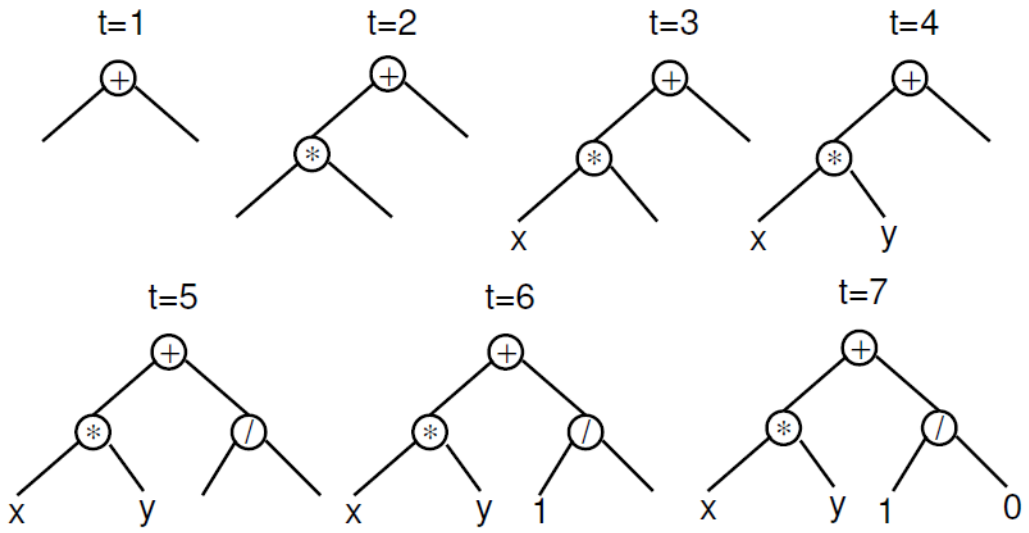


Figura 2.8: Formación de un árbol de altura 2 mediante el método *full* (t=tiempo) [57].

La Figura 2.8 muestra las etapas del proceso de construcción de un árbol de altura 2 utilizando el método *full*. En este ejemplo, los hijos de los nodos $*$ y / deben ser hojas, porque en caso contrario el árbol tendrá una altura mayor a 2. Por consiguiente, en $t = 3$, $t = 4$, $t = 6$ y $t = 7$ se escogen elementos del conjunto T (x, y, 1 y 0, respectivamente).

- ***Grow*:** En este método se pueden escoger aleatoriamente nodos tanto del conjunto F como del conjunto T hasta alcanzar la altura del árbol predefinida. Luego, al igual que en el método anterior, se termina utilizando sólo nodos del conjunto T. Cabe destacar que con este orden de creación no siempre se alcanza la altura máxima del árbol que fue establecida, debido a que se pueden utilizar nodos terminales antes de llegar a este punto.

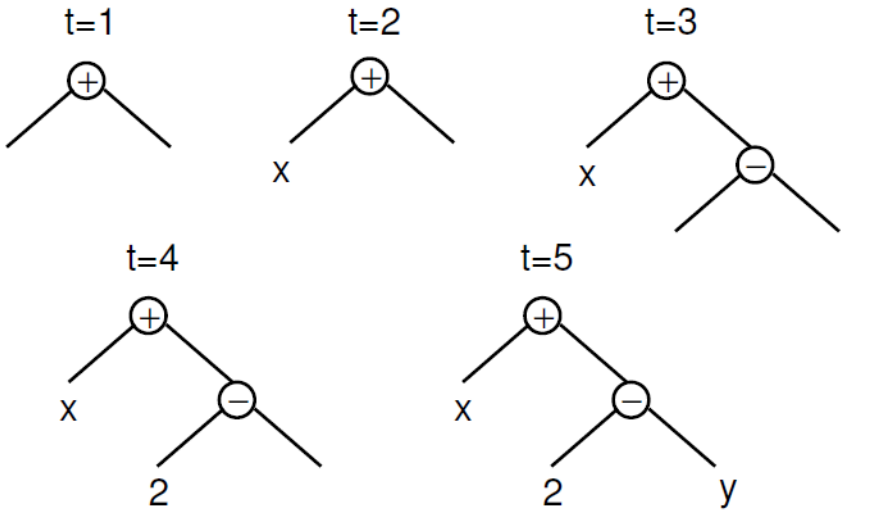


Figura 2.9: Formación de un árbol de altura máxima 2 mediante el método *grow* (t=tiempo) [57].

La Figura 2.9 ilustra el proceso de construcción de un árbol de altura máxima 2. Cómo el primer hijo del nodo raíz + corresponde a la variable x, que es un elemento terminal, esa rama del árbol no puede seguir creciendo, aun cuando no se ha alcanzado la altura límite. Por otro lado, los hijos del nodo $-$ deben ser elementos terminales para evitar sobrepasar esta restricción, por lo que en $t = 4$ y $t = 5$ se añaden los nodos 2 e y, respectivamente.

De esta forma se crean árboles de diversas formas y alturas, pero se debe tener en consideración que ambas características dependen fuertemente de la cardinalidad (número de elementos) de los conjuntos F y T. Si la cardinalidad del conjunto T es considerablemente mayor que la del conjunto F, el método *grow* formará mayoritariamente árboles de baja altura, independiente del límite impuesto. Por el contrario, si la cardinalidad del conjunto F es significativamente mayor que la del conjunto T, este modo se comportará de manera similar a *full*, en el sentido que producirá árboles cuya altura sea igual al límite establecido.

- ***Ramped half-and-half*:** Es utilizado debido a que ninguno de los dos métodos anteriores por si sólos son capaces de proporcionar una amplia gama de alturas y formas de los árboles generados. Consiste en que una mitad de la población es creada por el método *full* y la otra mitad por *grow*. Esto se hace utilizando un rango de límites de altura de los árboles, lo que ayuda a asegurar que la población inicial sea más variada.

2.3.4.5. Evaluación

Como ya se ha mencionado anteriormente, en el algoritmo de programación genética los individuos son evaluados para definir que tan buena solución al problema representa cada uno de ellos, haciendo posible la aplicación el principio de reproducción del más apto para la creación de la siguiente generación. La evaluación se lleva a cabo por la función de evaluación (en inglés *fitness function*), que asigna una calificación numérica de aptitud a cada individuo,

luego de resolver el problema planteado con la expresión matemática propuesta por dicho individuo y comparar el resultado obtenido con el deseado [99]. Esta función de evaluación debe ser proporcionada por el usuario y su elección tiene un rol fundamental en el desempeño del algoritmo [100].

En literatura se suele utilizar como función de evaluación al error cuadrático medio (ECM), representado en la Ecuación 2.14, donde $Y(X)$ es el valor de la variable dependiente, estimado por la expresión matemática propuesta por cada individuo, y $\hat{Y}(X)$ es el dato experimental observado.

$$ECM = \frac{1}{n} \cdot \sum_{i=1}^{n} (Y_i(X) - \hat{Y}_i(X))^2 \tag{2.14}$$

Además, se puede diseñar una función de evaluación que modifiquen a los individuos de una generación antes de calcular su fitness. Por ejemplo, se pueden optimizar los parámetros de las expresiones matemáticas representadas por cada uno de los individuos, siguiendo los principios evolutivos lamarckianos [81].

2.3.4.6. Selección

Los algoritmos de selección se utilizan para escoger cuáles individuos de la generación producirán descendencia y cuáles no. La selección es la fuerza motriz de GP, y cómo este algoritmo está basado en el principio de reproducción del más apto, se debe otorgar mayor probabilidad de reproducción a los individuos cuyo fitness sea mayor. Sin embargo, las opciones de reproducción de los individuos menos aptos no deben limitarse por completo para evitar la pérdida de diversidad de la población.

La presión de selección es el grado en que los mejores individuos son favorecidos en el proceso de selección, mientras mayor es la presión de selección, más se favorece a los individuos más aptos. Con mucha presión de selección la búsqueda terminará de manera prematura, mientras que con muy poca presión el proceso evolutivo será más lento de lo necesario [101]. Típicamente, se utiliza una presión de selección baja en etapas iniciales del algoritmo, permitiendo una amplia exploración en el espacio de búsqueda, y en las etapas finales es recomendable aumentar la presión de selección con la finalidad de estrechar el espacio de búsqueda. La selección dirige la búsqueda hacia regiones promisorias en el espacio de búsqueda [70].

Cuando un individuo es seleccionado, pasa a formar parte de la población intermedia, la cuál contiene a los individuos que producirán descendencia para la próxima generación luego de la aplicación de los operadores genéticos [102]. Por lo general, el tamaño de la población intermedia es igual al número de individuos en cada generación [103], por lo que los métodos de selección de individuos se repiten hasta completar el tamaño de la generación. Además, el mismo individuo puede repetirse más de una vez en la población intermedia [104]. Dos de los métodos usados son la selección por ruleta y el torneo:

- **Selección por ruleta:** Este método asume que la probabilidad de selección es proporcional al fitness del individuo. Si se consideran N individuos, cada uno caracterizado

por su fitness $w_i > 0$ (con $i = 1, 2, 3, ..., N$). La probabilidad de selección del i-ésimo individuo está dada por la Ecuación 2.15.

$$p_i = \frac{w_i}{\sum_{i=1}^{N} w_i} \qquad (i = 1, 2, 3, ..., N) \tag{2.15}$$

Como se muestra en la Figura 2.10, en una selección por ruleta se construye una línea de largo $\sum_{i=1}^{N} w_i$ a partir de N vectores consecutivos de longitud w_i (con $i = 1, 2, 3, ..., N$), que representan a cada uno de los N individuos. Luego, se genera un número aleatorio r entre 0 y la longitud de la línea ($0 < r < \sum_{i=1}^{N} w_i$) y lo ubica en el sector correspondiente, en el caso de este ejemplo w_3, de manera que se selecciona el individuo respectivo.

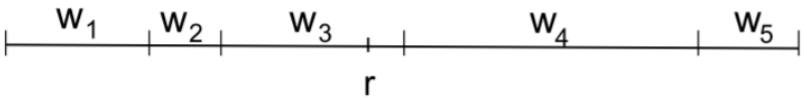


Figura 2.10: Ejemplo de distribución de los individuos para selección por ruleta [105].

- **Selección por torneo:** Este tipo de selección funciona de la siguiente manera: Primero se fija el tamaño de torneo, que corresponde número de individuos (t) que participará en cada torneo. Luego, se escogen aleatoriamente los t individuos y se selecciona al que tiene mayor fitness entre ellos para que forme parte de la población intermedia [106].

  El tamaño del torneo determina la presión de selección de este método. El individuo ganador de un torneo grande tendrá, en promedio, un mejor fitness que el ganador de un torneo más pequeño. Por lo tanto, cuando se busca mantener la diversidad de la población se realizan torneos de bajo tamaño, incluso existen los torneos binarios, que se realizan solamente entre dos individuos [101].

2.3.4.7. Operadores genéticos

Al igual que en GA, en programación genética existen tres operadores genéticos principales: entrecruzamiento o recombinación sexual (en inglés *crossover*), mutación y elitismo [74]. Estos operadores son utilizados para crear la descendencia de la población actual. Los individuos a partir de los cuáles se generará la nueva población son elegidos utilizando alguno de los métodos de selección descritos anteriormente.

- **Entrecruzamiento:** Este operador actúa sobre dos individuos padres y genera dos hijos. Consiste en seleccionar aleatoriamente un nodo de cada uno de los padres e intercambiar sus respectivos subárboles, como se ilustra en la Figura 2.11 en la que los padres intercambian las expresiones $x_3 + 6$ y $(x_1 - x_2) \cdot 2$.

  Es el operador genético principal y su función es proporcionar la mezcla de las soluciones para lograr su convergencia. La idea principal del cruce se basa en que, si se

toman dos individuos correctamente adaptados al medio y se obtiene una descendencia que comparta genes de ambos, existe la posibilidad de que los genes heredados sean precisamente los causantes de la bondad de los padres.

La frecuencia con que se aplica depende de un parámetro llamado tasa de entrecruzamiento. Esta tasa suele ser muy alta, debido a que este operador es el que permite que las posibles soluciones convergan hacia el óptimo, por lo general se fija en 90 % [57].

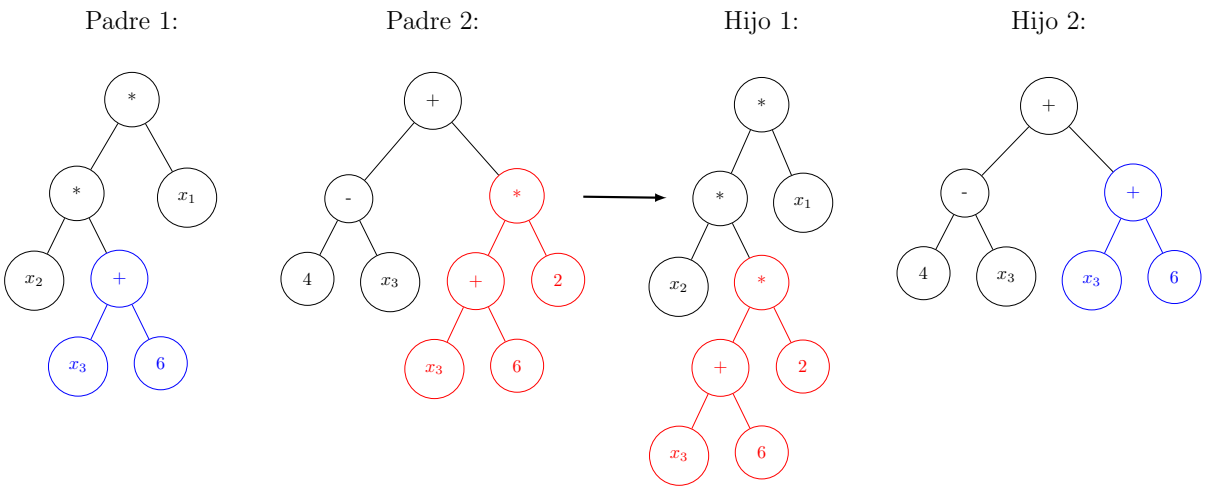


Figura 2.11: Ejemplo de entrecruzamiento. Se intercambian las expresiones $x_3 + 6$ y $(x_1 - x_2) \cdot 2$ entre Padre 1 y Padre 2.

- **Mutación:** Consiste en seleccionar un nodo aleatorio de un individuo y cambiar su subárbol por otro generado al azar, como se ejemplifica en la Figura 2.12, donde se selecciona aleatoriamente el nodo + y se cambia el subárbol que representa la expresión $x_3 + 6$ por la constante 7. La importancia de este operador radica en que aumenta la diversidad de la población, proporcionando un mecanismo para escapar de un óptimo local.

  La frecuencia con la que se aplica este operador es controlada por un parámetro llamado tasa de mutación. Por lo general, la tasa de mutación se fija entre un 5 % y un 20 %, debido a que una tasa mayor dificulta la convergencia de las soluciones hacia el óptimo y una tasa menor disminuye la variabilidad de la población, aumentando la probabilidad de que los individuos se estanquen en un mínimo local [107].

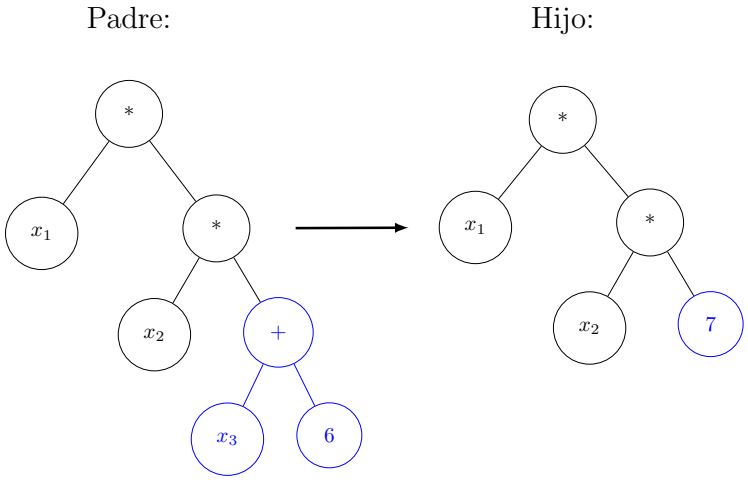


Figura 2.12: Ejemplo de mutación. Se cambia la expresión $x_3 + 6$ por la constante 7.

- **Elitismo:** También denominado reproducción asexual, corresponde a copiar una proporción de los individuos más aptos (con mejor fitness) en la siguiente generación sin realizarles cambios.

2.3.4.8. Parámetros importantes

A continuación, se describen los parámetros más importantes que pueden ser modificados por el usuario del algoritmo GP con la finalidad de mejorar su desempeño, ya sea disminuyendo el tiempo de ejecución o aumentando la calidad de la solución obtenida.

- **Tamaño de la población:** Es considerado como uno de los parámetros más importantes en el desempeño del algoritmo GP. Si el número de individuos creados en cada generación es muy pequeño es probable que no se logre encontrar un solución aceptable al problema planteado, mientras que si se tiene un tamaño de población muy grande el algoritmo tardará más tiempo del necesario en encontrar dicha solución [108].

  Es necesario que cada generación tenga el número de individuos suficiente para garantizar su variabilidad genética. Por lo general, el tamaño de la población se encuentra entre 100 y 500 individuos, pudiendo ser mayor dependiendo de la complejidad del problema que se quiere resolver, pero es poco frecuente que se superen los 1000 individuos [87].

- **Número de generaciones:** Este parámetro corresponde a uno de los criterios de término de GP. Cuando se llega al número de generaciones fijado por el usuario, el algoritmo deja de ejecutarse y la solución al problema planteado corresponde al mejor individuo generado hasta ese momento.

  Es común que el número de generaciones se encuentre entre 50 y 100, ya que el costo computacional y el tiempo que tarda en ejecutarse el algoritmo por más generaciones suele ser muy alto en comparación con el beneficio obtenido [87]. En caso de no obtener

los resultados esperados es recomendable aumentar el tamaño de la población antes que el número de generaciones [109].

- **Probabilidad de cruce**: Como ya se mencionó anteriormente, la probabilidad de cruce, o tasa de entrecruzamiento, determina la frecuencia con que se aplica el operador genético de entrecruzamiento a los individuos de una población intermedia para crear la nueva generación. La probabilidad de cruce se suele fijar en un 90 %, debido a que este operador genético es el que permite la convergencia de las soluciones hacia un óptimo [57].

- **Probabilidad de mutación:** Este parámetro determina la frecuencia con que se emplea el operador genético de mutación en los individuos de una población intermedia. Por lo general, se utilizan valores relativamente bajos, entre el 5 % y el 20 %, debido a que las mutaciones introducen variabilidad genética en la población, evitando que las soluciones se estanquen en un óptimo local, pero si ocurren con mucha frecuencia pueden dificultar su convergencia [107].

- **Profundidad máxima de árbol:** Al fijar una profundidad máxima para los árboles sintácticos se evita su crecimiento excesivo, limitando la complejidad de las soluciones al problema planteado. Si este parámetro es muy bajo puede que no sea posible formar la expresión matemática de la solución óptima, sobre todo en los problemas que requieren expresiones más complejas. Además, si se impide la formación de este tipo de expresiones, se limita la variabilidad de la población. En cambio, si la profundidad máxima del árbol es muy alta, los individuos tienen a crecer innecesariamente, y por lo general, tardan más tiempo en ser evaluados [110]. Por lo tanto, este parámetro depende de la complejidad del problema y suele limitarse entre 8 y 50 [57].

2.3.4.9. Criterios de término

El algoritmo de programación genética se ejecuta hasta que se cumpla una condición o criterio de término. Existen dos tipos de condiciones que resultan en la finalización del algoritmo: el criterio generacional y el criterio de desempeño. El primero se relaciona con alcanzar el número máximo de generaciones definido por el usuario y el segundo se cumple cuando el fitness de la población alcanza alguna condición establecida, que en principio es muy estricta: que todos los individuos tengan fitness 0. Sin embargo, para casos en que no se espera encontrar una solución tan exacta, como cuando se busca identificar un modelo matemático a partir de datos experimentales que presentan error, se suele fijar una condición menos estricta para terminar el algoritmo [87].

Por lo general, la manera utilizada para finalizar el algoritmo es mediante el criterio generacional, y es común que en algunos problemas no se incluyan criterios de desempeño, simplemente se analizan los resultados luego de iterar por el número máximo de generaciones.

2.3.4.10. Problemas del algoritmo GP

En las últimas décadas ha habido un amplio desarrollo en la teoría y aplicaciones del algoritmo de programación genética. Sin embargo, aún existen problemas que ocurren frecuentemente y que deben ser solucionados para obtener el máximo potencial de este algoritmo,

especialmente para su aplicación para la solución de problemas complejos del mundo real. A continuación, se describen dichos problemas y qué técnicas se utilizan para disminuir su efecto.

- **Sobreajuste:** Uno de los objetivos más importantes de los algoritmos de regresión simbólica es encontrar soluciones que se desempeñen bien no sólo en los casos conocidos, o de entrenamiento, sino que también proporcionen predicciones satisfactorias de los casos desconocidos. Lo anterior se conoce como generalización, y la falta de generalización se denomina sobreajuste [111]. El sobreajuste ocurre cuando una solución se desempeña bien en los casos de entrenamiento, pero mal en los casos de prueba o en cualquier otro caso desconocido. Esto indica que las relaciones subyacentes entre los datos no fueron aprendidas, y en su lugar, se aprendieron relaciones que existen exclusivamente en el caso de entrenamiento [112].

  Cuando se seleccionan modelos matemáticos basándose únicamente en su fitness, el algoritmo tenderá a escoger expresiones más complejas que se sobreajustan al conjunto de datos de entrenamiento, absorbiendo los errores aleatorios. Por lo tanto, para evitarlo, se recomienda seleccionar modelos no sólo dependiendo de su fitness, si no que también considerando su complejidad [113]. Por lo tanto, la función de evaluación utilizada es altamente responsable de la evolución de modelos matemáticos generalizados [100]. Algunas de las funciones de evaluación que pueden utilizarse para incluir la complejidad del modelo en la selección son el criterio de error de predicción final de Akaike (FPE) [114], el criterio de información de Akaike (AIC) [115], el criterio de información bayesiano (BIC) [116] o la minimización del riesgo estructural (SRM) [117] [118]. Otra opción corresponde a implementar un algortimo GP multiobjetivo, que maximice el fitness mientras minimiza la complejidad de las soluciones [119].

  Sin embargo, existen estudios que demuestran que el sobreajuste puede ocurrir incluso cuando se controla la complejidad de los individuos [120], y que hay casos en los que algunas expresiones más simples pueden tener un mayor error de generalización [121]. Por lo tanto, existen otras técnicas que, en vez de considerar la complejidad de los individuos, utilizan en cada generación un subconjunto de datos experimentales distinto para evaluar a los individuos, como la técnica de muestreo aleatorio (RST) [112] o la de muestreo intercalado [122]. De esta manera, a través de las generaciones sólo se mantendrán en la población aquellos individuos que tengan un buen desempeño con varios conjuntos de datos distintos. Otra alternativa es utilizar dos conjuntos de datos en cada generación, uno para entrenar a los individuos y otro para validarlos, como ocurre en el *backwarding* [123] o en la utilización de funciones canario [124].

- **Bloat:** También llamado *bloating*, se puede definir como el crecimiento de las soluciones propuestas por el algoritmo sin que ocurran mejoras significativas o proporcionales en términos de fitness [57]. Es decir, es la tendencia de GP de evolucionar hacia soluciones que son innecesariamente largas o complejas. Cuando esto ocurre, el número de nodos promedio de la población aumenta incluso cuando el fitness de la mejor solución se estanca. Esto puede enlentecer la búsqueda de la solución debido a que, por lo general, las expresiones matemáticas más complejas requieren de más recursos computacionales,

especialmente en la etapa de evaluación [125].

El primer método utilizado para evitar que este problema ocurra corresponde a limitar la profundidad de árbol de los individuos [87]. Además, se pueden utilizar para disminuir el *bloating* las técnicas mencionadas que consideran la complejidad de los individuos en el control del sobreajuste. También existen técnicas de edición de los individuos que tienen la finalidad de eliminar información redundante dentro de su código. El *snipping* es un ejemplo de este tipo de métodos [126]. Otro recurso ampliamente utilizado corresponde al operador de ecualización, que controla la distribución de tamaños de los individuos en la población [127].

Capítulo 3

Metodología

3.0.1. Metodología general

Como se muestra en la Figura 3.1, la metodología propuesta para identificar funciones o expresiones matemáticas desconocidas en modelos diferenciales consta de cuatro etapas. Primero, se deben obtener datos experimentales del problema estudiado. Es necesario medir dos conjuntos de datos bajo condiciones iniciales distintas, uno de ellos se utilizará para la ejecución del algoritmo de programación genética (datos de entrenamiento) y el otro en el proceso de validación del mejor modelo generado por el algoritmo (datos de validación). Preferentemente, los datos deben tener información sobre el sistema tanto en su etapa dinámica como en el estado estacionario y estar tomados equiespaciadamente.

Una vez obtenidos los datos, se ejecuta el algoritmo GP con la finalidad de encontrar la estructura matemática del bloque faltante. Para esto es necesario diseñar la función de evaluación como se muestra en la Sección 3.0.1.1. Luego de ejecutar el algortimo, se optimizan los parámetros de la mejor solución propuesta por y finalmente se valida este modelo con la finalidad de medir su capacidad de generalización.

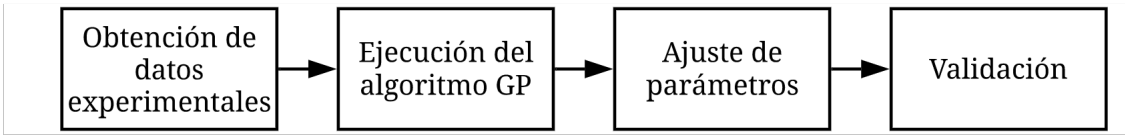


Figura 3.1: Metodología propuesta para identificar bloques faltantes en sistemas diferenciales.

El proceso de validación consiste en simular el modelo obtenido con las condiciones iniciales utilizadas para obtener los datos experimentales de validación y comparar los resultados con ese conjunto de datos. De esta manera, se puede comprobar si la mejor solución propuesta por el algoritmo GP corresponde a un modelo generalizado que es aplicable para diversas condiciones iniciales o si es un caso de sobreajuste de los datos experimentales de entrenamiento.

3.0.1.1. Diseño de la función de evaluación

La función de evaluación diseñada calcula el fitness de cada individuo generado por el algoritmo. Esta función tiene como variables de entrada al individuo, los datos experimentales, las estructuras de las ecuaciones diferenciales que forman el sistema y puede incluir una parte conocida del bloque faltante buscado. Como se esquematiza en la Figura 3.2, la función objetivo primero extrae la expresión matemática representada por el individuo, para luego incorporarla con la parte conocida para formar el bloque faltante. Luego, se construye el sistema de ecuaciones diferenciales al reemplazar la expresión del bloque faltante en las ecuaciones conocidas que lo incluyen. Este sistema de ecuaciones es resuelto numéricamente para calcular el fitness del individuo como el error cuadrático medio entre la trayectoria de las variables obtenida mediante la simulación numérica del modelo y los datos experimentales disponibles.

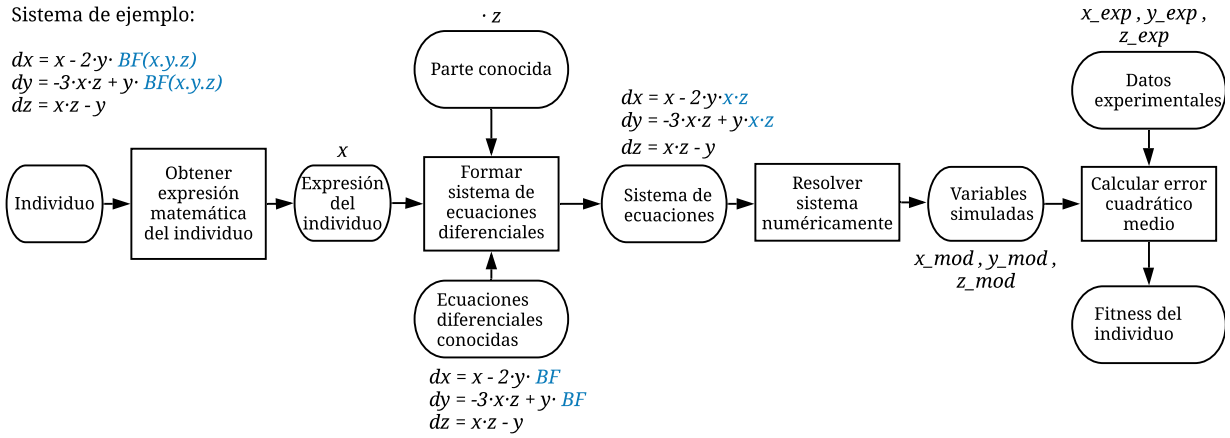


Figura 3.2: Funcionamiento general de la función objetivo diseñada.

En la Figura 3.2 se ejemplifican los pasos seguidos por la función de evaluación diseñada para la identificación de un bloque desconocido en un sistema de ecuaciones diferenciales compuesto por las Ecuaciones 3.1, 3.2 y 3.3. La expresión $BF(x, y, z)$ corresponde a la función o bloque faltante que se desea identificar.

$$\frac{dx}{dt} = x - 2 \cdot y \cdot BF(x, y, z) \tag{3.1}$$

$$\frac{dy}{dt} = -3 \cdot x \cdot z + y \cdot BF(x, y, z) \tag{3.2}$$

$$\frac{dz}{dt} = x \cdot z - y \tag{3.3}$$

Como se muestra en el esquema de la Figura 3.2, la función objetivo diseñada requiere que se conozca la estructura de las ecuaciones diferenciales y su relación con el bloque faltante. Por lo tanto, se utiliza un enfoque semi-empírico para identificar a la función desconocida, ya que se necesita aportar información sobre las ecuaciones diferenciales de todas las variables del sistema, pero no es necesario tener conocimiento previo o realizar supuestos sobre la

expresión matemática que se quiere identificar.

Además, se puede incluir información adicional sobre este bloque mediante la variable denominada *parte conocida*. En el ejemplo, se sabe de antemano que contiene la expresión $\cdot z$, por lo que se incluye como parte conocida y es añadida a la expresión matemática propuesta por cada individuo (en el ejemplo x, formando el bloque faltante $x \cdot z$). En la Figura 3.3, se muestra en detalle el proceso de formación del sistema de ecuaciones, donde las ecuaciones que describen a las variables x e y incluyen el bloque faltante y la que describe la variable z no.

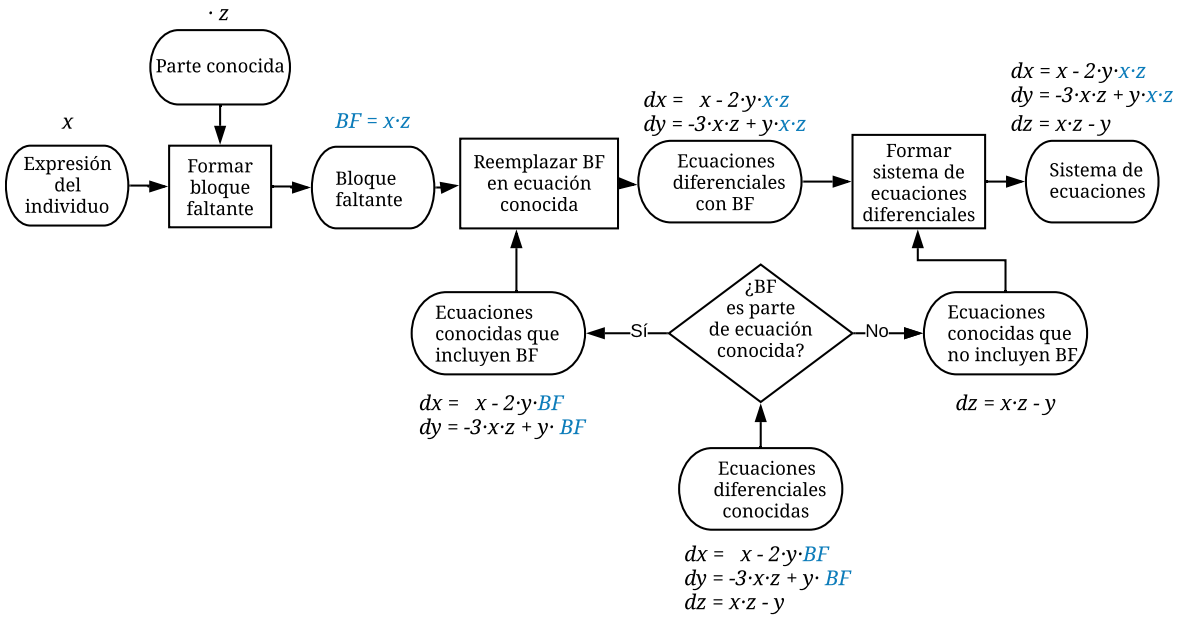


Figura 3.3: Detalle de la formación del sistema de ecuaciones diferenciales.

Para la resolución numérica de los sistemas de ecuaciones diferenciales se utilizó el algoritmo basado en el método de Rosenbrock de orden 2 por su buen desempeño en la resolución de problemas de tipo stiff [128]. Los problemas stiff son aquellos que tienen componentes que varían lentamente y otros que varían rápidamente [129]. Es altamente probable que el algoritmo proponga individuos que generen sistemas de ecuaciones stiff, más aún cuando se incluye la función potencia (a^b). Los métodos de Runge-Kutta no son adecuados para la resolución de este tipo de problema y suelen tomar significativamente más tiempo que el algoritmo seleccionado [130]. Por lo tanto, esta elección permite evitar que GP se estanque en la resolución del sistema de ecuaciones generado por un individuo, disminuyendo entonces su tiempo total de ejecución. La desventaja de esta elección es que el método de Rosenbrock es menos exacto en la resolución de problemas no stiff que los métodos de Runge-Kutta, pero esta diferencia, por lo general, no es tan significativa con respecto tiempo que se ahorra cuando se resuelven problemas stiff.

Un posible problema de la ejecución del algoritmo GP con la función de evaluación diseñada es que es probable que existan individuos que posean estructuras matemáticas útiles para la construcción de la solución óptima, pero que tengan parámetros poco adecuados que

produzcan que su fitness sea muy alto (error cuadrático medio alto), aumentando la probabilidad de que sean eliminados en la etapa de selección. Esto puede producir que dichas estructuras necesarias para la construcción del modelo se pierdan en el proceso [131]. Una alternativa para evitar que esto ocurra es la implementación de la búsqueda local, que consiste en optimizar los parámetros de todos los individuos antes de la etapa de evaluación [132].

La desventaja de incorporar la búsqueda local a GP es que se aumenta considerablemente el gasto computacional del algoritmo, y en consecuencia, el tiempo que tarda en ejecutarse. Lo anterior debido a que se debe resolver el sistema de ecuaciones formado por cada individuo tantas veces como iteraciones del método de optimización de parámetros se realicen. A esto se le debe sumar el tiempo que tarda el método de optimización en el proceso de búsqueda de los parámetros. Por lo tanto, si el tiempo es limitado, el algoritmo debe ejecutarse con un tamaño de población significativamente menor, lo que disminuye la diversidad de la población, reduciendo la probabilidad de encontrar la solución óptima. En dicho caso la utilización del método propuesto originalmente es más recomendable debido a la posibilidad de ejecutarse con tamaño de poblaciones significativamente mayores. Una buena opción es incluir la optimización de parámetros a un pequeño porcentaje de la población, de esta manera el tiempo de ejecución no aumenta tanto como en el caso en que se ajustan los parámetros de todos los individuos y se disminuye la probabilidad de que se extingan expresiones útiles por tener parámetros mal ajustados. Otro posible problema de la implementación de la búsqueda local es que aumenta la probabilidad de que se produzca sobreajuste a los datos experimentales de entrenamiento [133].

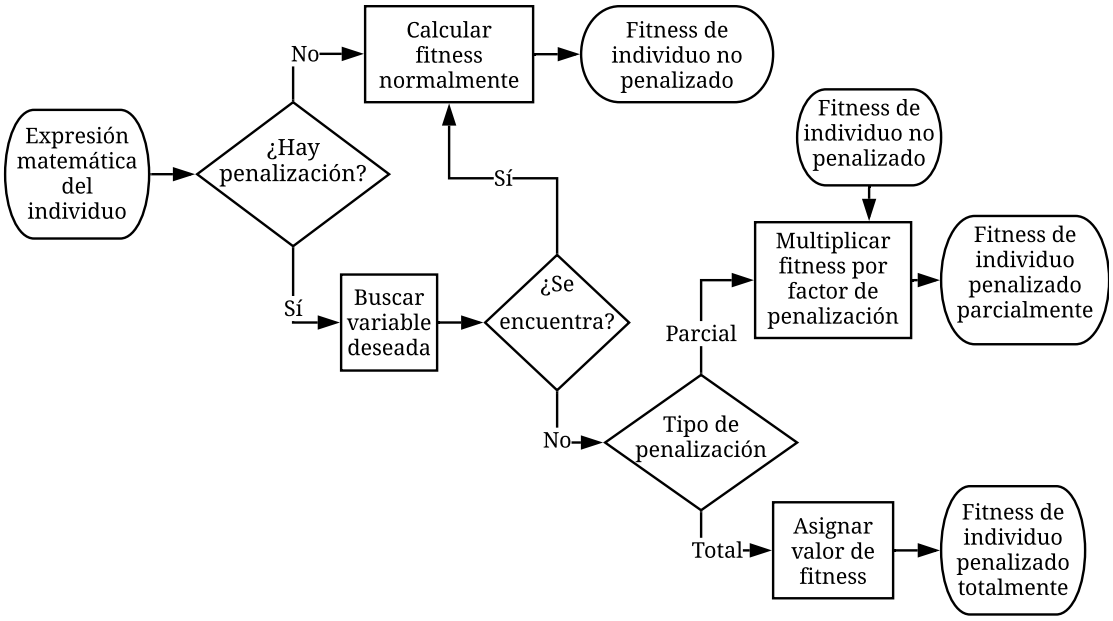


Figura 3.4: Funcionamiento de la penalización al fitness de un individuo por la ausencia de una variable.

Basándose en la penalización por complejidad de la solución propuesta [134], en este trabajo se propone incluir la opción de penalizar a aquellos individuos cuya expresión matemática no incluya a alguna variable que el usuario considera debe ser utilizada. El funcionamiento de esta característica se muestra en la Figura 3.4. La penalización puede ser total o parcial:

cuando es total se le asigna un valor de fitness definido por el usuario a todos los individuos que no cuenten con la variable deseada, mientras que cuando es parcial se calcula el fitness del individuo normalmente y luego se multiplica por un factor de penalización, también definido por el usuario.

Esta penalización permite guiar la búsqueda de la solución basándose en el nivel de conocimiento que el usuario tiene sobre el sistema. Se presume que la ventaja de la penalización total es que reduce el tiempo de ejecución del algoritmo al no resolver el sistema de ecuaciones diferenciales para los casos en que no se cuenta con la variable deseada, pero la desventaja es que asigna el mismo valor de fitness a todos los individuos penalizados, por lo que no es posible compararlos entre sí. Es probable que alguno de los individuos penalizados tenga expresiones que son necesarias para alcanzar la solución óptima, pero que esas expresiones se pierdan debido a que no habría manera de advertir que el individuo en cuestión representa una mejor solución que el resto de los individuos penalizados, por lo tanto no tendría prioridad sobre ellos para dejar descendencia.

Se plantea la penalización parcial como una solución al problema de diferenciar a los individuos penalizados, ya que tendrán fitness distintos entre sí. De todas maneras, se debe escoger un factor de penalización adecuado, ya que de ser muy alto se tendría un comportamiento similar al de la penalización total, y si es muy bajo no se lograría el efecto deseado de orientar la búsqueda hacia individuos que posean la variable deseada. La desventaja con respecto a la penalización total es que tarda más tiempo en ejecutarse debido a que se deben solucionar los sistemas de ecuaciones que representan a todos los individuos. Sin embargo, esta no es una desventaja con respecto al caso en que no se aplica penalización, ya que en esa situación se resuelve el mismo número de sistemas de ecuaciones diferenciales.

3.0.2. Estrategia para evaluar viabilidad de metodología propuesta

Para evaluar la viabilidad de la metodología propuesta, se siguieron los pasos descritos en la Figura 3.5. Primero se seleccionaron casos de estudio correspondientes a modelos matemáticos diferenciales no lineales que describen fenómenos biológicos. Luego, para cada caso de estudio se escogió un bloque faltante, que corresponde a la función o expresión matemática que no se conoce y se busca identificar con el algoritmo de programación genética. Después, se implementó el algoritmo GP para seguir la metodología propuesta para intentar identificar expresiones matemáticas que representen al bloque faltante escogido y posteriormente evaluar su desempeño.

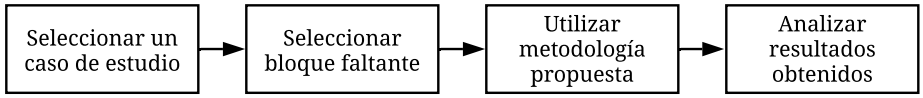


Figura 3.5: Estrategia utilizada para evaluar viabilidad de metodología propuesta.

Los casos de estudio utilizados en el procedimiento de evaluación de la metodología son sintéticos, es decir, los conjuntos datos experimentales utilizados fueron generados a partir de

simulaciones de cada uno de los modelos matemáticos seleccionados y no provienen de experimentos realizados realmente. El objetivo del primer caso fue estudiar el funcionamiento del algoritmo, verificar su capacidad de proponer funciones para el bloque faltante y estudiar el impacto que tienen en su desempeño diversos factores: el tamaño de la población, el número máximo de generaciones, el nivel de conocimiento previo que se tiene sobre el fenómeno y el error en los datos experimentales. Por otro lado, el segundo caso de estudio tuvo como objetivo verificar la aplicabilidad de la metodología para la identificación de expresiones matemáticas desconocidas en un modelo matemático de un sistema biológico que se asemeje a alguno de la industria de procesos.

También fue necesario implementar el algoritmo de programación genética. Se utilizó como base GPLAB, que es una caja de herramientas de programación genética para MATLAB [135]. Las funciones de evaluación genéricas de GPLAB sólo permiten identificar modelos matemáticos integrales, por lo que fue necesario diseñar una función que proponga expresiones matemáticas para modelos diferenciales. Para obtener los datos experimentales se resolvieron numéricamente los sistemas de ecuaciones diferenciales que describen cada uno de los casos de estudio utilizando el método de Dormand-Prince (Runge-Kutta de cuarto y quinto orden), y como se representa en la Ecuacion 3.4, para obtener los datos experimentales con error (Y_{error}) se generó un vector de números generados aleatoriamente con distribución de probabilidad uniforme continua (E) y se sumó al vector de datos experimentales generados sintéticamente (Y).

$$Y_{error} = Y + E \tag{3.4}$$

En la Tabla 3.1, se muestran los parámetros comunes utilizados para la ejecución del algoritmo GP en todas las repeticiones de ambos casos de estudio. El número de generaciones, tamaño de individuo y variables incluidas en el conjunto de terminales varía dependiendo del caso de estudio.

Tabla 3.1: Parámetros utilizados para la ejecución del algoritmo GP.

Parámetro	**Valor o criterio**
Probabilidad de entrecruzamiento	0,9
Probabilidad de mutación	0,1
Método de selección	Torneo
Método de inicialización	*Ramped half-and-half*
Funciones	+, -, *, /
Terminales	Constantes: 1, 5, 10, 100, 1000
Profundidad máxima de árbol	10
Función de evaluación	Error cuadrático medio

El algoritmo GP opera sintácticamente, por lo que para que un parámetro de un individuo sea modificado este debe encontrarse explícitamente en la expresión matemática del individuo en cuestión. Por ejemplo, en la expresión mostrada en 3.5, GP puede utilizar operadores genéticos sobre los parámetros 0,5, 2 y 10, pero no puede hacerlo sobre las constantes 1 y -1 que multiplican implícitamente a las variables $S \cdot X$ y S, respectivamente. En la fase de ajuste de parámetros de los mejores individuos se incluyeron explícitamente este tipo de pa-

rámetros, como se muestra en el ejemplo 3.5, donde p_1, p_2, p_3, p_4 y p_5 son los parámetros que serán optimizados, siendo p_3 y p_5 constantes que se encontraban implícitas en la expresión matemática, por lo que no eran modificables por el algoritmo GP. El ajuste de parámetros se realizó minimizando el error cuadrático medio a través del método Nelder-Mead [136].

$$\frac{(0,5 \cdot S + 2 \cdot X + S \cdot X)}{(10 - S)} \longrightarrow \frac{(p_1 \cdot S + p_2 \cdot X + p_3 \cdot S \cdot X)}{(p_4 + p_5 \cdot S)} \tag{3.5}$$

Capítulo 4

Resultados

4.1. Caso de estudio 1: Cultivo de microorganismos en reactor batch

Este caso de estudio tiene como finalidad estudiar el efecto del tamaño de la población, del número de generaciones, del nivel de conocimiento previo que se tiene sobre el sistema y del error en los datos experimentales en el desempeño del algoritmo de programación genética. Por lo tanto, se escogió un modelo no lineal relativamente simple que es útil para un primer acercamiento a la identificación de bloques faltantes en sistemas diferenciales biológicos, que posteriormente puede ser complejizado al incluir la síntesis de algún producto, efectos de inhibición por sustrato o por producto o el efecto de la de variación de condiciones de operación (temperatura o pH, por ejemplo).

4.1.1. Modelo estudiado

El primer caso de estudio corresponde al crecimiento de un microorganismo en un reactor batch operando a temperatura y pH constante. La tasa específica de crecimiento del microorganismo sigue el modelo de Monod y no se considera la muerte del microorganismo ni el consumo de sustrato para la mantención de las células, de manera que el fenómeno estudiado está descrito por las Ecuaciones 4.1 y 4.2.

$$\frac{dX}{dt} = \mu_{max} \cdot (\frac{S}{K_S + S}) \cdot X \tag{4.1}$$

$$\frac{dS}{dt} = -\frac{1}{Y_{X/S}} \cdot \mu_{max} \cdot (\frac{S}{K_S + S}) \cdot X \tag{4.2}$$

4.1.2. Obtención de los datos experimentales

Los parámetros utilizados en este caso de estudio se muestran en la Tabla 4.1. Para obtener los datos experimentales se resolvió el problema de valor inicial compuesto por las ecuaciones diferenciales 4.3a y 4.3b, y las condiciones iniciales 4.3c y 4.3d, utilizando el método numérico

de Dormand-Prince [137].

Tabla 4.1: Parámetros utilizados para simular el caso de estudio 1.

Parámetro	μ_{max} (h^{-1})	K_S (g/L)	$Y_{X/S}$ (g/g)
Valor	0,4	0,3	1,25

$$\begin{cases} \dfrac{dX}{dt} = \dfrac{0.4 \cdot S}{S + 0,3} \cdot X & (4.3a) \\ \dfrac{dS}{dt} = -0,8 \cdot \dfrac{0.4 \cdot S}{S + 0.3} \cdot X & (4.3b) \\ X|_{t=0} = 1 & (4.3c) \\ S|_{t=0} = 30 & (4.3d) \end{cases}$$

Para la validación de los modelos propuestos por el algoritmo GP se utilizaron nuevos datos experimentales, que fueron obtenidos al simular las ecuaciones diferenciales 4.3a y 4.3b con nuevas condiciones iniciales: $X_0 = 1,5$ (g/L) y $S_0 = 75$ (g/L). Los conjuntos de datos, tanto de entrenamiento como de validación, generados se muestran en el gráfico de la Figura 4.1, en el que se puede apreciar que el lapso de tiempo simulado fue de 12 horas, lo que permite tener información sobre el fenómeno tanto en su etapa dinámica como en el estado estacionario.

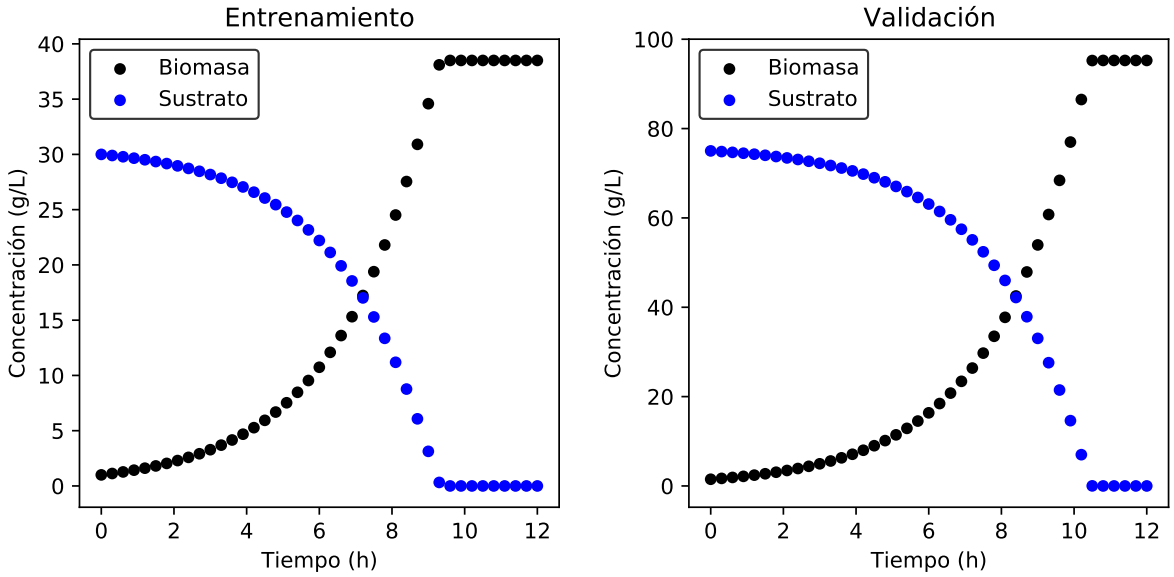


Figura 4.1: Gráficos de los datos experimentales de simulados para el entrenamiento y validación de modelos matemáticos en el caso de estudio 1.

4.1.3. Bloque faltante estudiado

Como se muestra en la Ecuación 4.4, en este caso de estudio el bloque faltante (BF) corresponde a todo el lado derecho de la ecuación que describe la tasa de crecimiento celular. Por lo tanto, para utilizar la metodología propuesta y diseñar la función de evaluación, se debe tener conocimiento sobre la relación entre el bloque faltante y la ecuación diferencial que describe al sustrato, mostrada en la Ecuación 4.6, donde EI corresponde a la expresión del bloque faltante propuesta por un individuo generado por el algortimo GP.

$$BF = \frac{0.4 \cdot S \cdot X}{S + 0,3} \tag{4.4}$$

$$\frac{dX}{dt} = EI \tag{4.5}$$

$$\frac{dS}{dt} = -\frac{1}{Y_{X/S}} \cdot EI \tag{4.6}$$

Además, se debe conocer el valor del rendimiento de generación de biomasa a partir del sustrato ($Y_{X/S}$), que puede ser calculado a partir de las concentraciones iniciales y finales de células y de sustrato como se muestra en la Ecuación 4.7, donde X_0 y S_0 corresponden a las concentraciones iniciales, y X_{ss} y S_{ss} a las concentraciones en el estado estacionario.

$$Y_{X/S} = \frac{X_0 - X_{ss}}{S_{ss} - S_0} \tag{4.7}$$

Como en este caso de estudio el bloque faltante tiene una estructura relativamente sencilla, con la finalidad de evitar el crecimiento innecesario de las soluciones obtenidas (*bloat*), se decidió penalizar asignando un fitness de $1 \cdot 10^{16}$ a aquellos individuos cuyos árboles tengan más de 15 nodos. Esta penalización se realiza de manera similar a penalización total mostrada en la Figura 3.4, pero en vez de buscar una variable deseada se chequea si el inidivuo en cuestión supera o no el límite de nodos establecido.

4.1.4. Estudio del efecto del tamaño de la población

Para estudiar el efecto del tamaño de la población en el desempeño del algoritmo GP se realizaron 50 repeticiones para distintos números de individuos por generación (100, 150, 250, 500, 750 y 1000). El objetivo de la etapa de ejecución del algoritmo GP es que el mejor individuo propuesto genere una expresión (EI) con la estructura mostrada en la Ecuación 4.8, para que en la etapa de ajuste de parámetros se obtengan las constantes de la Ecuación 4.4 ($p_1 = 0,4$, $p_2 = 1$ y $p_3 = 0,3$).

$$EI = \frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3} \tag{4.8}$$

En la Tabla 4.2 se presenta la expresión matemática propuesta por el individuo de mejor fitness para cada tamaño de población, junto con dicha expresión con los parámetros ajustados. Además se presenta el fitness del individuo previo al ajuste de parámetros (Fitness GP), luego del ajuste (Fitness ajustado) y el fitness de la etapa de validación (Fitness validación).

Los parámetros ajustados las expresiones para los tamaños de población 150, 250 y 750 fueron divididos por el parámetro ajustado de la variable S del denominador de la expresión (en el ejemplo de la Ecuación 4.8, los parámetros serían divididos por el valor de p_2), para facilitar su comparación con los parámetros del bloque faltante mostrado en la Ecuación 4.4.

El fitness GP se calcula sobre expresiones matemáticas que no tienen sus parámetros ajustados, por lo que es probable que el individuo con mejor fitness GP no represente a la mejor solución obtenida en las 50 repeticiones para cada tamaño de población. Como se evidencia en la Tabla 4.2, el mejor individuo para el tamaño de población 100 tiene mejor fitness GP que el mejor individuo para el tamaño de población 150, pero luego del ajuste de parámetros el segundo tiene mejor desempeño que el primero. Lo anterior demuestra un problema de la ejecución del algoritmo utilizando la función de evaluación diseñada, ya que existen individuos que tienen una estructura de expresión matemática con el potencial de ser una buena solución, pero que pueden ser descartados en la etapa de selección por tener mal fitness debido a que sus parámetros no están ajustados. Un ejemplo de esto ocurre con el mejor individuo del tamaño de población 250, cuya expresión matemática tiene exactamente la misma estructura que el bloque faltante objetivo (Ecuación 4.8), pero tiene peor fitness GP que los mejores individuos de tamaño de población 500, 750 y 1000, que tienen expresiones distintas a las de dicho bloque faltante objetivo.

Tabla 4.2: Individuos con mejor fitness GP luego de realizadas las 50 repeticiones del algoritmo para cada tamaño de población.

Tamaño población	**Expresión GP**	**Fitness GP**	**Expresión Ajustada**	**Fitness Ajustado**	**Fitness Validación**
100	$0,0010 \cdot S \cdot X^2 + 1$	2,489	$0{,}0015 \cdot S \cdot X^2 + 0,642$	$0,416$	1928
150	$\frac{5 \cdot S + S \cdot X + 1}{5 \cdot S + 1}$	6,876	$\frac{-0,001 \cdot S + 0,4 \cdot S \cdot X + 0,0031}{1 \cdot S + 0,322}$	$3,63 \cdot 10^{-5}$	$0,008$
250	$\frac{2 \cdot S \cdot X}{5 \cdot S + 2}$	0,0122	$\frac{0,4 \cdot S \cdot X}{1 \cdot S + 0,3}$	$1{,}74 \cdot 10^{-8}$	$4{,}42 \cdot 10^{-6}$
500	$\frac{-X + 10 \cdot S \cdot X - 10}{25 \cdot S}$	0,0094	$\frac{-0,646 \cdot X + 13,1 \cdot S \cdot X + 10,5}{33,5 \cdot S}$	0,0045	2,210
750	$\frac{50 \cdot S \cdot X}{100 \cdot S + 5 \cdot X + S^2}$	$0,062$	$\frac{0,396 \cdot S \cdot X}{1 \cdot S + 0,0077 \cdot X - 1,08 \cdot 10^{-6} \cdot S^2}$	$1,73 \cdot 10^{-8}$	$0,784$
1000	$\frac{99 \cdot S \cdot X - 50}{250 \cdot S}$	0,0083	$\frac{107 \cdot S \cdot X - 1,45}{270 \cdot S}$	0,0079	19723

Por lo anterior, en la Tabla 4.2 se puede observar que los individuos con mejor fitness GP no necesariamente representan modelos matemáticos generalizados del fenómeno estudiado, lo que se refleja en el pobre fitness de validación de los mejores individuos para los tamaños de población 100 y 1000 (1928 y 19723, respectivamente). Seleccionar al mejor individuo por su fitness ajustado tampoco garantiza que este represente a un modelo generalizado. Por este motivo, la etapa de validación es fundamental para descartar aquellos resultados que en un principio parecen ser buenas soluciones al problema planteado, pero que realmente se tratan de casos de sobreajuste. Por lo tanto, es recomendable escoger como la mejor solución para

cada tamaño de población a aquella que tiene mejor fitness de validación, de esta manera se verifica que corresponde a un modelo generalizado del fenómeno estudiado.

En la Tabla 4.3 se presenta el individuo con mejor desempeño en la etapa de validación para cada tamaño de población. Como se puede observar, las soluciones con mejor fitness de validación difieren de aquellas con mejor fitness GP para prácticamente todos los tamaños de población, con la excepción de 250, y en todos los casos el fitness de validación que permite garantizar que se tratan de modelos generalizados, por lo que efectivamente es más conveniente seleccionar a los individuos basándose en su desempeño en esta etapa. Además se observa que, para este caso de estudio en específico, se logran buenos resultados a pesar del problema de ajuste de parámetros mencionado anteriormente, pero es posible que tenga un efecto más notorio en la resolución de casos más complejos.

Tabla 4.3: Individuos con mejor fitness validado para cada tamaño de población.

Tamaño población	**Expresión GP**	**Fitness Validación**	**Expresión Ajustada**	**Fitness GP**	**Fitness Ajustado**
100	$\frac{200 \cdot S + 200 \cdot X + 200 \cdot S \cdot X}{-200 \cdot S + 20001}$	$3{,}23 \cdot 10^{-6}$	$\frac{1{,}02 \cdot 10^{-5} \cdot S - 3{,}41 \cdot 10^{-8} \cdot X + 0{,}4 \cdot S \cdot X}{1 \cdot S + 0{,}3}$	22,92	$9{,}18 \cdot 10^{-8}$
150	$\frac{S \cdot X}{2 \cdot S + X + 1}$	$2{,}97 \cdot 10^{-6}$	$\frac{0{,}4 \cdot S \cdot X}{1 \cdot S - 1{,}58 \cdot 10^{-7} \cdot X + 0{,}3}$	7,173	$2{,}09 \cdot 10^{-8}$
250	$\frac{2 \cdot S \cdot X}{5 \cdot S + 2}$	$4{,}61 \cdot 10^{-6}$	$\frac{0{,}4 \cdot S \cdot X}{1 \cdot S + 0{,}3}$	0,0122	$1{,}74 \cdot 10^{-8}$
500	$\frac{10 \cdot X + 101 \cdot S \cdot X}{255 \cdot S + 51}$	$5{,}12 \cdot 10^{-6}$	$\frac{2{,}31 \cdot 10^{-6} \cdot X + 0{,}4 \cdot S \cdot X}{1 \cdot S + 0{,}3}$	0,0119	$1{,}76 \cdot 10^{-8}$
750	$\frac{10 \cdot X + 101 \cdot S \cdot X}{25 \cdot S + 50}$	$5{,}15 \cdot 10^{-6}$	$\frac{6{,}71 \cdot 10^{-7} \cdot X + 0{,}4 \cdot S \cdot X}{1 \cdot S + 0{,}3}$	0,0124	$1{,}77 \cdot 10^{-8}$
1000	$\frac{20 \cdot S \cdot X}{10 \cdot S - 11 \cdot X + 1000}$	$5{,}06 \cdot 10^{-6}$	$\frac{0{,}4 \cdot S \cdot X}{1 \cdot S - 4{,}88 \cdot 10^{-7} \cdot X + 0{,}3}$	4,956	$1{,}78 \cdot 10^{-8}$

Como se puede observar en la Tabla 4.3, el ajuste de parámetros también permite corregir la estructura de las expresiones matemáticas propuestas como solución al aproximar a cero los parámetros de las variables extra que no aportan al fitness del individuo. De todas maneras, es importante realizar un test de significancia para verificar que dichos parámetros no tienen un efecto significativo en el modelo planteado por cada individuo. Al realizar dicho test, se comprobó que todos los parámetros mencionados pueden ser despreciados, por lo que todas expresiones de la Tabla 4.3 tienen realmente la misma estructura y parámetros que el resultado obtenido para el tamaño de población 250, cuyo análisis estadístico se presenta en la Tabla 4.4. En el Anexo A.1 se muestra en detalle el proceso de eliminación de parámetros no significativos.

Tabla 4.4: Análisis estadístico de la mejores solución obtenida en el estudio del tamaño de población.

Expresión	**Parámetro**	**Intervalo de confianza (95 %)**	**p-value**
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3}$	$p_1 = 0,4$	$\pm 6,49 \cdot 10^{-4}$	0
	$p_2 = 1$	$\pm 1,69 \cdot 10^{-3}$	0
	$p_3 = 0,3$	$\pm 9,95 \cdot 10^{-4}$	0

Para estudiar el efecto del tamaño de la población en el desempeño del algoritmo, se consideran como modelos matemáticos generalizados, y por ende soluciones aceptables, todos aquellos casos en que tanto el fitness ajustado como el fitness de validación es menor que 0,01. Los individuos cuyo fitness ajustado es menor que 0,01 y su fitness de validación es mayor que 0,01 se consideran sobreajustados; y los individuos cuyo fitness ajustado es mayor que 0,01 se consideran casos de mal ajuste a los datos experimentales de entrenamiento. El criterio de fitness ajustado y fitness de validación menor que 0,01 para considerar que un modelo es generalizado fue impuesto arbitrariamente, basándose en que los datos experimentales no tienen error y en que la función de evaluación corresponde al error cuadrático medio, por lo que pequeñas diferencias (menores que uno) entre los datos experimentales y las simulaciones de los modelos obtenidos se hacen aún más pequeñas. Por lo tanto, el umbral de 0,01 significa que, en promedio, los puntos de los modelos aceptables tienen una diferencia de 0,1 con los datos experimentales.

De todas maneras, el fitness umbral impuesto podría ser incluso más exigente para este caso de estudio en específico. En los gráficos de la Figura 4.2 se muestra que pasaría si dicho umbral cambiara de 0,01 a $1 \cdot 10^{-4}$. En la Figura 4.2a se presenta la distribución de fitness ajustado de los casos generalizados, y se evidencia que más del 80 % de estos casos generalizados tienen un fitness ajustado menor que $1 \cdot 10^{-7}$, mientras que menos de un 5 % sería considerado caso de mal ajuste (región verde del gráfico) si se cambiara el fitness umbral a $1 \cdot 10^{-4}$. El gráfico de la Figura 4.2b presenta la distribución de fitness de los casos sobreajustados, y se aprecia que aproximadamente un 80 % de ellos tienen un fitness ajustado mayor que $1 \cdot 10^{-4}$, por lo que se considerarían mal ajustados si se cambiara el umbral. Finalmente, la Figura 4.2c presenta la distribución del fitness de validación de los casos generalizados en función de su fitness ajustado, y cerca del 85 % estos casos tienen un fitness de validación menor que $1 \cdot 10^{-4}$. En síntesis, en caso de cambiar el fitness umbral desde 0,1 a $1 \cdot 10^{-4}$, un 5 % y un 10 % de los casos generalizados pasarían a ser mal ajustados y sobreajustados, respectivamente; y un 80 % de los casos sobreajustados pasarían a ser de mal ajuste.

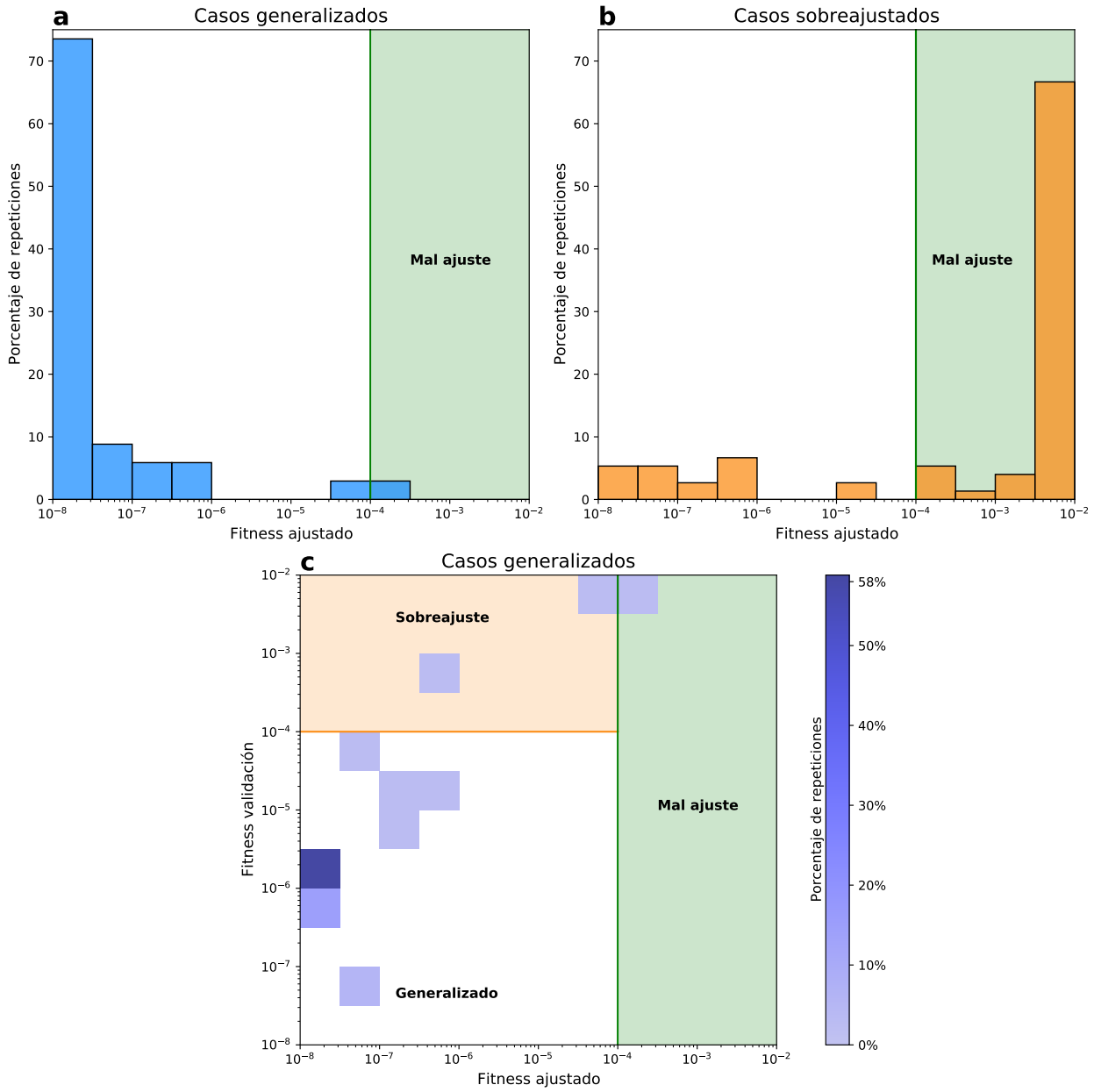


Figura 4.2: (a) Distribución del fitness ajustado de todos los casos generalizados. (b) Distribución del fitness ajustado de todos los casos sobreajustados. (c) Mapa de calor del fitness de validación de los casos generalizados en función del fitness ajustado. En (a), (b) y (c) la región verde muestra casos que se considerarían de mal ajuste si se cambiara el fitness umbral de $1 \cdot 10^{-3}$ a $1 \cdot 10^{-4}$ y la región amarilla corresponde a los casos que se considerarían sobreajustados al cambiar el fitness umbral.

El gráfico de la Figura 4.3 muestra la distribución de casos generalizados, sobreajustados y de mal ajuste para los distintos tamaños de población. Este gráfico confirma la importancia de la etapa de validación porque cerca de un tercio de los modelos cuyo fitness ajustado es menor que 0,01 (sobreajustados y generalizados) corresponden a modelos generalizados, mientras que el resto son casos sobreajustados. Además, se observa que la ejecución del

algoritmo GP utilizando la función de evaluación diseñada en este caso de estudio permite identificar con éxito un modelo matemático generalizado en por lo menos una repetición de cada uno de lo tamaños de población. El mayor porcentaje de repeticiones exitosas (31 %) ocurre al ejecutar el algoritmo con poblaciones de 500 individuos por generación. Para los tamaños de población menores que 500 individuos el porcentaje de éxito es notoriamente menor (2 %, 2 % y 6 % para 100, 150 y 250 individuos por generación, respectivamente). Además, el porcentaje de éxito también es menor para poblaciones de mayor tamaño (15 % y 17 % para 750 y 1000 individuos, respectivamente), produciéndose un aumento de casos de mal ajuste.

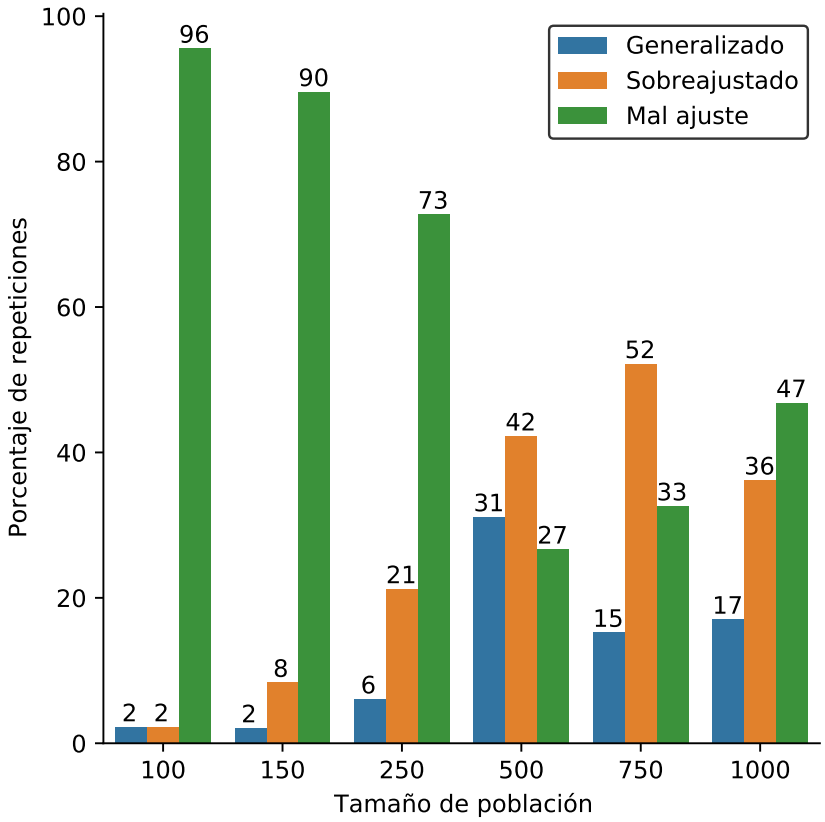


Figura 4.3: Gráfico de la distribución de casos generalizados, sobreajustados y de mal ajuste para los distintos tamaños de población.

Cuando GP se ejecuta con tamaños de población pequeños la exploración del espacio de búsqueda es más acotada, por lo que es más probable que se estanque en óptimos locales o que simplemente no alcance a converger en 50 generaciones. Esto explica el aumento del porcentaje de casos generalizados entre los tamaños de población de 100 y 500. Para poder explicar el descenso en el porcentaje de casos generalizados para los tamaños de población 750 y 1000, primero se deben introducir los conceptos de individuos equivalentes e individuos inútiles. Como se mencionó en la Sección 2.3.4.1, expresiones matemáticas equivalentes pueden ser representadas por más de un árbol sintáctico distinto. El algoritmo GP implementado en la herramienta GPLAB identifica a los individuos a partir de su representación en árbol y no de su expresión matemática, por lo que, al ejecutar el algoritmo es muy difícil evitar que aparezcan individuos que representan expresiones matemáticas equivalentes con árboles

distintos, estos son los individuos equivalentes. Se presume que los individuos equivalentes pueden aparecer simplemente debido a la aleatoriedad con que se forman los individuos o porque la población está convergiendo hacia un mínimo, ya sea local o global, lo que hace que ciertas expresiones matemáticas se utilicen en mayor proporción al crear nuevos individuos. Por otro lado, se consideran individuos inútiles aquellos que no cumplen con la restricción de nodos máximos impuesta en la función de evaluación, por lo que se les asigna un valor de fitness de $1 \cdot 10^{16}$. Debido a su alto fitness las expresiones matemáticas representadas por estos individuos no pueden ser comparadas con las de los individuos que si cumplen con la restricción de nodos.

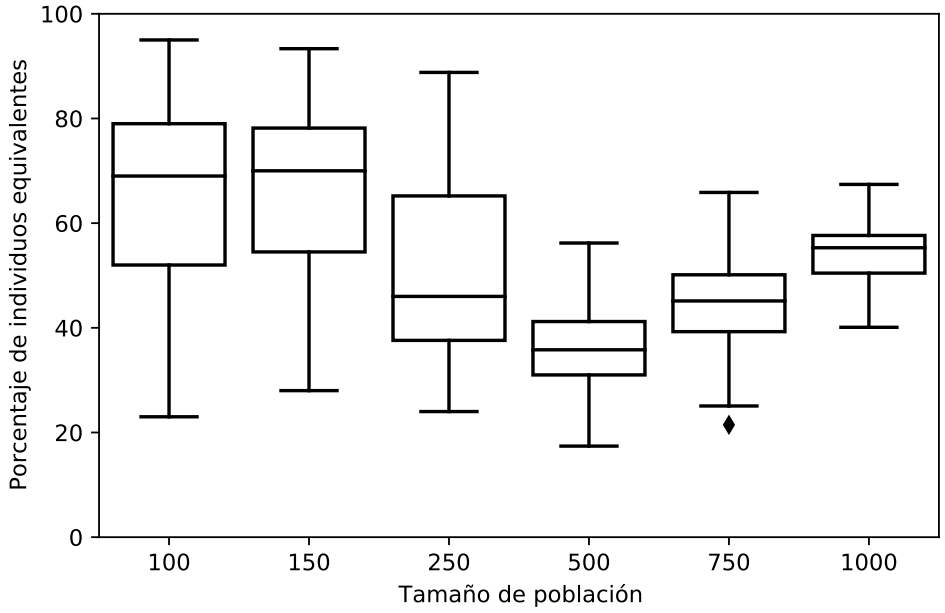


Figura 4.4: Gráfico de la distribución del porcentaje de individuos equivalentes en la última generación de las distintas repeticiones en función del tamaño de población.

En la Figura 4.4 se muestra la distribución del porcentaje de individuos equivalentes en la última generación en cada una de las 50 repeticiones realizadas para los distintos tamaños de población. Se considera sólo la última generación de cada iteración porque con GPLAB sólo es posible obtener información de todos los individuos de esta generación. En gráfico de la Figura 4.4 se observa que para los tamaños de población 100, 150 y 250 el rango de porcentajes de individuos equivalentes varía entre alrededor del 20 % y 90 %. La mediana para 100 y 150 individuos se encuentra cerca del 75 %, lo que puede ser una muestra de convergencia a un mínimo local. Además, los tamaños de población 750 y 1000 tienen un mayor porcentaje de individuos equivalentes que el tamaño de población 500.

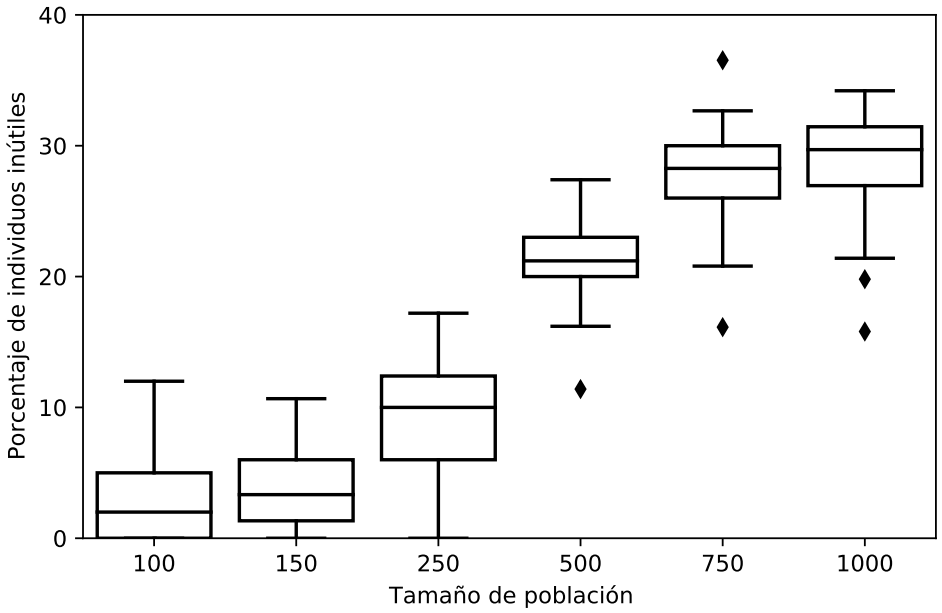


Figura 4.5: Gráfico de la distribución del porcentaje de individuos inútiles en la última generación de las distintas repeticiones en función del tamaño de población.

La Figura 4.5 presenta la distribución del porcentaje de individuos inútiles presentes en la última generación de cada una de las 50 repeticiones realizadas para cada tamaño de población. Se observa que este porcentaje tiende a aumentar con el tamaño de población, llegando a ser alrededor del 25 % y el 30 % para los tamaños de población 750 y 1000, respectivamente. Esto se debe a que lo que se debe a que mientras mayor es el tamaño de población, el algoritmo va a tender a evolucionar hacia soluciones más complejas, en este caso con mayor número de nodos, por lo que habrá más individuos a los que se le asigna un fitness de $1 \cdot 10^{16}$ [138].

Al eliminar los individuos equivalentes, dejando sólo uno de ellos, y los individuos inútiles se obtiene la población efectiva de individuos distintos en la última generación. En la Figura 4.6 se presenta la distribución del tamaño de la población efectiva en la última generación en las repeticiones realizadas para cada tamaño de población, y se puede apreciar que la mediana del número de individuos efectivos es menos de la mitad de la población original para todos los casos. Además, la disminución de casos generalizados para los tamaños de población 750 y 1000 que se evidenció en la Figura 4.3 se puede explicar con la disminución de la población efectiva para estos casos en comparación con 500 individuos.

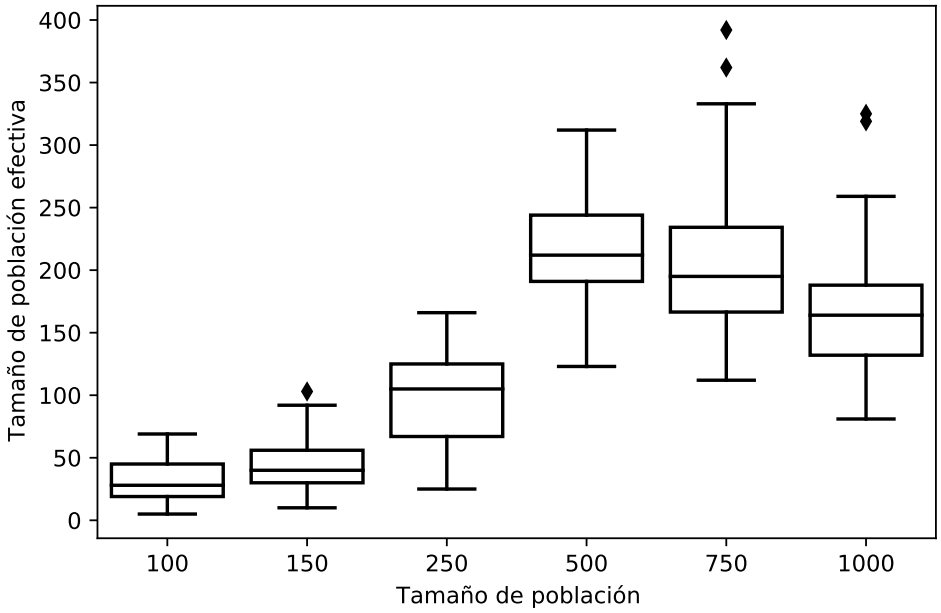


Figura 4.6: Gráfico de la distribución del tamaño de población efectiva en la última generación de las distintas repeticiones en función del tamaño de población.

Por lo tanto, para mejorar el desempeño del algoritmo se debería modificar su funcionamiento para que el tamaño de población efectiva sea el mismo que el tamaño de población original. Para lograrlo, se necesita evitar la aparición de individuos equivalentes y de individuos inútiles. Para el caso de los individuos equivalentes, al momento de crear un nuevo individuo, se podría verificar que la expresión matemática que este representa no sea equivalente a la de algún otro individuo de la población, descartándolo en caso de no cumplir con dicha condición. Para el caso de los individuos inútiles la solución sería rechazar a los individuos que sobrepasen el número máximo de nodos establecidos en vez de penalizarlos, pero una mejor opción es simplemente no imponer esta penalización, puesto que ya se cuenta con el límite de profundidad máxima de árbol, y en caso de querer evitar el crecimiento innecesario de las soluciones se puede disminuir dicho número. Para ambas modificaciones se debe modificar el funcionamiento del algoritmo en GPLAB, pero podría hacerse de la misma manera que este rechaza a los individuos cuyos árboles superan el límite de profundidad establecido.

Otro aspecto que se debe considerar es el tiempo que tarda en ejecutarse cada repetición del algoritmo con los distintos tamaños de población. En la Figura 4.7, se puede apreciar que el tiempo de ejecución del algoritmo aumenta junto con el tamaño de población, porque se tienen que evaluar cada vez más individuos resolviendo el sistema de ecuaciones diferenciales que representa cada uno de ellos. La importancia del tiempo de ejecución es que el desempeño de cada tamaño de población, medido en porcentaje de repeticiones exitosas, debe justificar el tiempo que el algoritmo tarda en llegar a estos resultados.

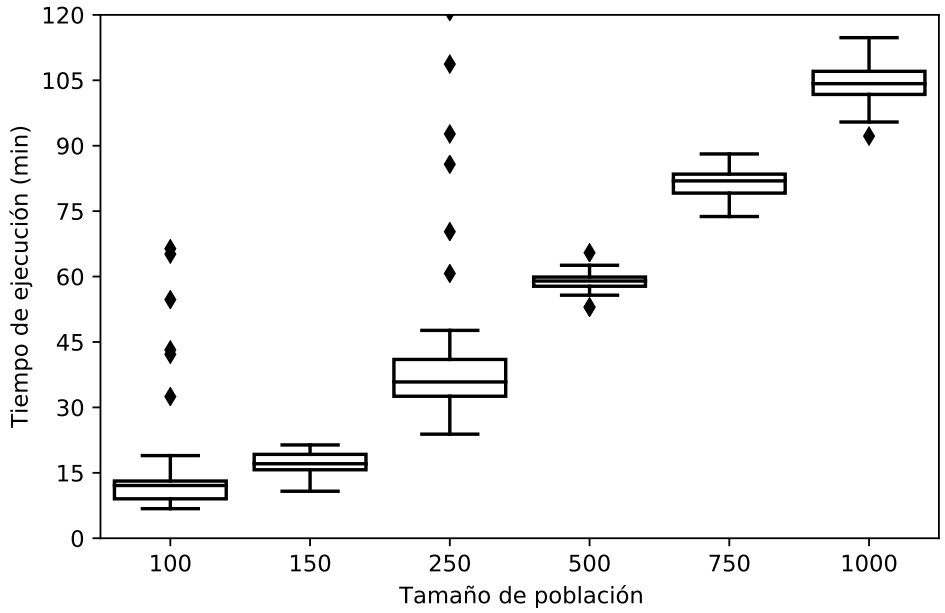


Figura 4.7: Gráfico de la distribución del tiempo de ejecución de las distintas repeticiones del algoritmo GP en función del tamaño de la población.

El tamaño de población con mejor relación entre porcentaje de repeticiones de éxito y tiempo de ejecución por repetición es 500 individuos por generación (31 % repeticiones exitosas, cada una de estas tomando alrededor de 60 minutos). El tamaño de población que lo sigue es 250 individuos por generación (6 % de éxito tardando alrededor de 40 minutos por repetición), que se demora aproximadamente 1,5 veces menos, pero tiene un quinto del porcentaje de éxito que tiene 500 individuos por generación. Incluso si lo tamaños de población de 750 y 1000 individuos tuvieran un mayor porcentaje de repeticiones exitosas que 500, este debería compensar el tiempo extra que tarda cada repetición. En caso contrario, sería más conveniente ejecutar el algoritmo con 500 individuos por una mayor cantidad de repeticiones, dónde cada repetición adicional tardaría una hora aproximadamente (Figura 4.7). Entonces, se necesitarían alrededor de 50 horas para realizar 50 repeticiones más.

4.1.5. Estudio del efecto del número de generaciones

Se estudió el efecto del número de generaciones en el algoritmo de programación genético al ejecutarlo por 25, 50, 75 y 100 generaciones, con un tamaño de población de 500 individuos y realizando 50 repeticiones para cada caso. El gráfico de la Figura 4.8 muestra, para cada número de generaciones, el porcentaje de repeticiones en que se obtuvieron modelos generalizados, sobreajustados y mal ajustados.

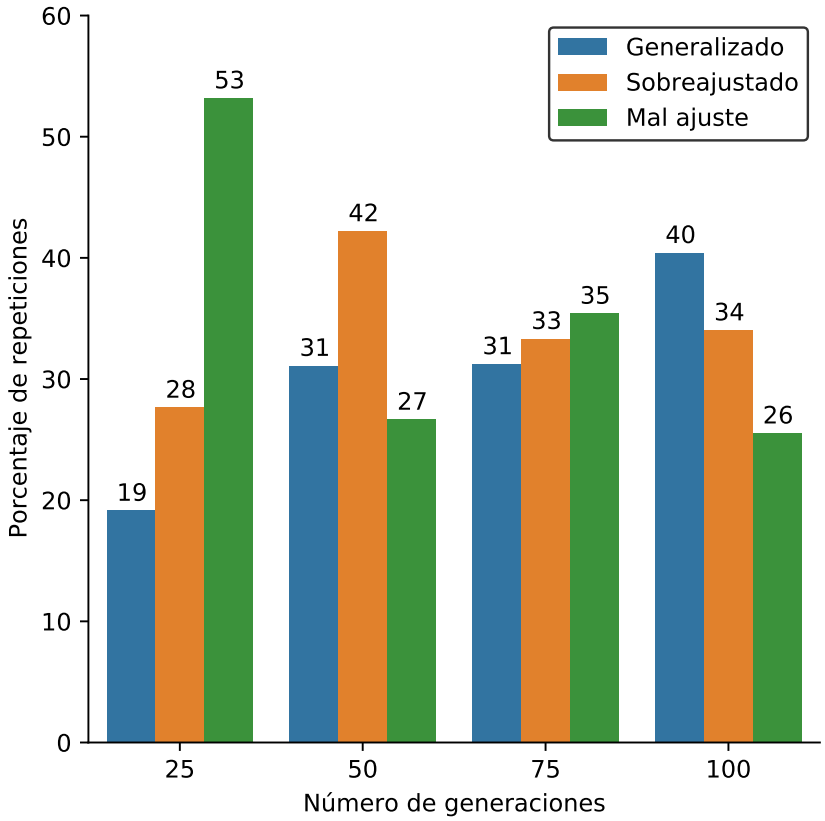


Figura 4.8: Gráfico de la distribución de casos generalizados, sobreajustados y de mal ajuste para los distintos números de generaciones.

Como se observa en la Figura 4.8, si el algoritmo se ejecuta por un mayor número de generaciones el porcentaje de repeticiones exitosas tiende a aumentar. Sin embargo, se debe tener el cuenta el tiempo que se demora en cada caso, presentado en el gráfico de la Figura 4.9, donde se puede observar que ejecutar el algoritmo por 50 y 100 generaciones tarda aproximadamente 2 y 3,5 veces más que hacerlo por 25 generaciones. Por lo tanto, para este caso en específico, el aumento en el porcentaje de éxito a lo largo de las generaciones no compensa el tiempo extra que se demora la ejecución del algoritmo, y según la tendencia mostrada en la Figura 4.8, si el tiempo es una limitante para la identificación del modelo matemático se podrían realizar más repeticiones de 25 generaciones y lograr más casos exitosos. Esto es consistente con la recomendación encontrada en literatura de priorizar aumentar el tamaño de la población antes que el número de generaciones cuando no se obtienen los resultados esperados, debido a que los mayores cambios en el mejor individuo ocurren en las primeras generaciones [109].

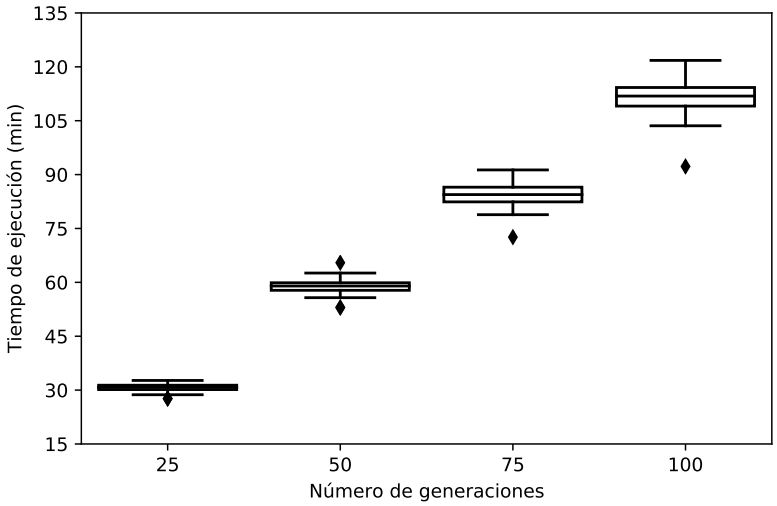


Figura 4.9: Gráfico del tiempo de ejecución (min) de las repeticiones realizadas para los distintos números de generaciones.

En la Figura 4.10 se muestra la distribución de los cambios de mejor individuo ocurridos luego de realizar las 50 repeticiones por 100 generaciones. En el gráfico de la 4.10a se evidencia que, a medida que aumenta el número de generaciones, cada vez ocurren menos cambios de mejor individuo. Además, en la Figura 4.10b se presenta la distribución de la cantidad de cambios ocurridas cada 25 generaciones y se puede apreciar con mayor claridad que después de la generación 50 suceden sólo el 16 % del total de cambios. Este comportamiento se condice con la recomendación presentada anteriormente de priorizar realizar repeticiones por un menor número de generaciones porque la evolución del mejor individuo se hace menos probable a medida que pasan las generaciones.

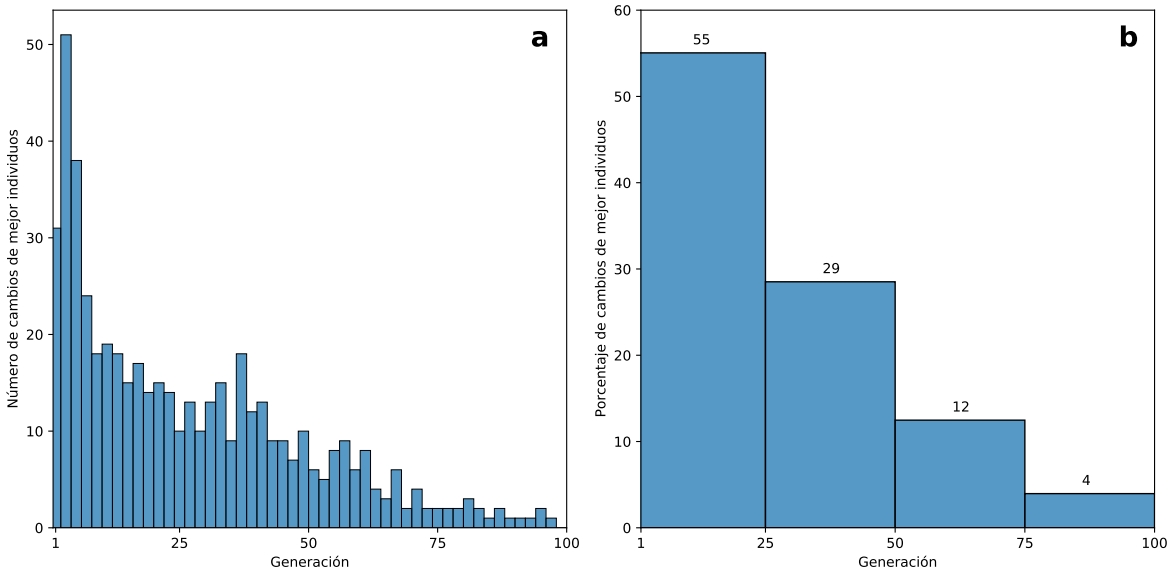


Figura 4.10: (a) Gráfico del número de cambios de mejor individuo que ocurre en cada generación. (b) Gráfico del porcentaje de cambios de individuos que ocurre en cada intervalo de 25 generaciones.

4.1.6. Efecto de la inclusión de una parte conocida del bloque faltante

Se estudió el efecto que tiene en el desempeño del algoritmo GP la inclusión de la variable $\cdot X$ como parte conocida de la función que se desea identificar. Se realizaron 50 repeticiones de 50 generaciones cada una, y el número de individuos por generación corresponde a aquél que tuvo peor desempeño en el estudio del efecto del tamaño de la población (100 individuos), de esta manera se puede apreciar de mejor manera el efecto de la inclusión de una parte conocida. El gráfico de la Figura 4.11 muestra el porcentaje de repeticiones generalizadas, sobreaajustadas y mal ajustadas del resultado obtenido previamente para este número de individuos sin parte conocida (en el Gráfico 100 ind sin PC), para el caso en que se añade la expresión $\cdot X$ como parte conocida (100 ind PC = X) y para el mejor resultado obtenido previamente, que corresponde a la ejecución del algoritmo por 100 generaciones con un tamaño de población de 500 individuos (en el Gráfico 500 ind 100 gen).

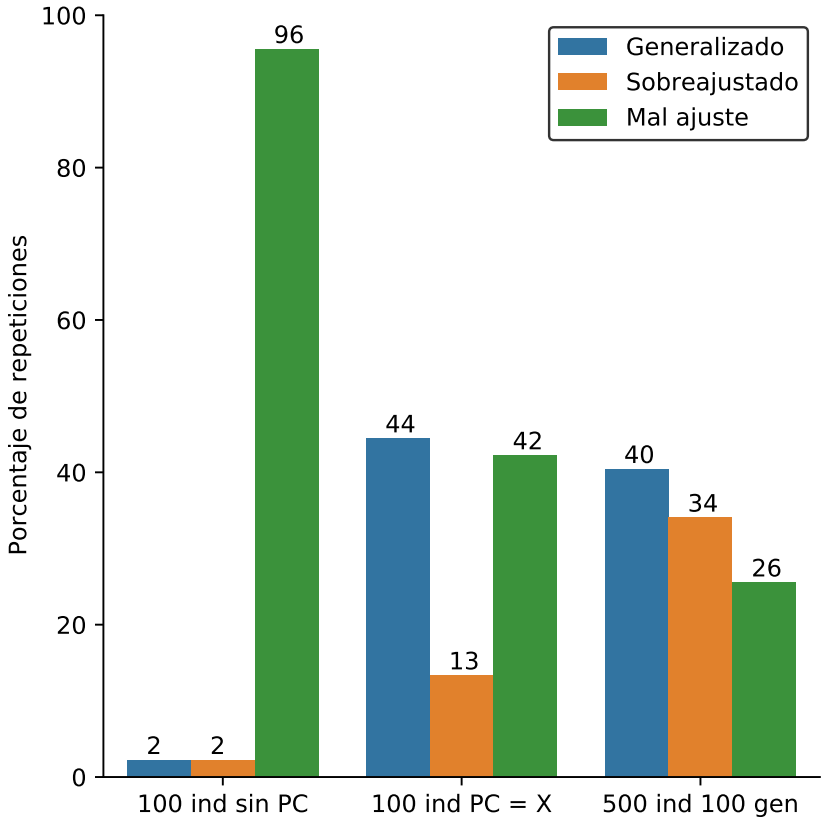


Figura 4.11: Distribución de casos generalizados, sobreajustados y de mal ajuste para las repeticiones de 50 generaciones con 100 individuos, con y sin parte conocida, y para las repeticiones realizadas con 500 individuos por 100 generaciones.

Como se aprecia en el gráfico de la Figura 4.11, al incluir la expresión $\cdot X$ como parte conocida se logra un porcentaje de repeticiones exitosas (44 %) muy superior al resultado obtenido sin esta herramienta (2 %). Además se puede observar que este porcentaje de repeticiones exitosas incluso es superior al mejor resultado obtenido previamente (40 %, cuando

se iteró el algoritmo con 500 individuos por 100 generaciones). Como ya se mencionó en la Sección 4.1, la parte conocida se combina con la expresión matemática generada por cada individuo para formar la expresión propuesta para el bloque faltante. Por lo general, mientras más simple es una expresión matemática es más fácil para el algoritmo GP identificarla. Entonces el gran aumento en el porcentaje de repeticiones exitosas cuando se agrega la parte conocida ocurre porque la expresión matemática que debe ser identificada por los individuos (EI) en GP se simplifica. En este caso, al incluir $\cdot X$ como parte conocida (PC) la EI original (Ecuación 4.8) cambia como se muestra en la Ecuación 4.9, donde EI' corresponde a la nueva expresión que se debe identificar.

$$EI' = \frac{EI}{PC} = \frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3} \cdot \frac{1}{X} = \frac{p_1 \cdot S}{p_2 \cdot S + p_3} \tag{4.9}$$

Además, en el gráfico de la Figura 4.12 se puede apreciar que el tiempo de ejecución del algoritmo GP prácticamente se mantiene con la inclusión de una parte conocida, debido a que la cantidad de cálculos que deben realizarse es la misma en ambos casos, la única diferencia es que la expresión propuesta por cada individuo es modificada antes ser evaluada. Entonces, en comparación con los resultados obtenidos previamente, al agregar la parte conocida no sólo se obtiene un mayor porcentaje de repeticiones exitosas, sino que también se logra en mucho menos tiempo.

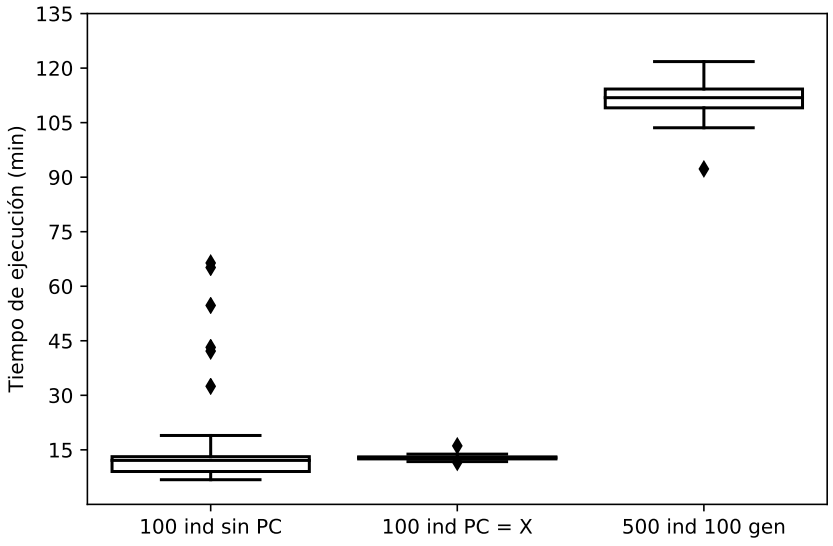


Figura 4.12: Gráfico del tiempo de ejecución de GP para las repeticiones de 50 generaciones con 100 individuos, con y sin parte conocida, y para las repeticiones realizadas con 500 individuos por 100 generaciones.

Por lo tanto, el grado de conocimiento que se tenga sobre el sistema estudiado es muy útil para mejorar el desempeño del algoritmo mediante la inclusión de una parte conocida del bloque faltante, que tiene el potencial de generar resultados aceptables con mayor frecuencia iterando el algoritmo GP con un tamaño de población menor y por un menor número de generaciones, lo que se traduce en menos tiempo de ejecución. En este caso en específico la

introducción de la expresión $\cdot X$ tiene sentido porque es sabido que tanto la tasa de crecimiento de levaduras como la tasa de consumo de sustrato dependen de la concentración de células. Específicamente, cuando no se considera la muerte celular ni el consumo de sustrato para el mantenimiento de los microorganismos, ambas tasas dependen de $\mu \cdot X$. Entonces, cuando se introduce la parte conocida antes mencionada el usuario busca que el algoritmo identifique una función para la tasa de crecimiento específico del microorganismo (μ).

Esta herramienta podría ser utilizada incluso sin tener conocimiento previo sobre el problema por resolver, en casos en que no se obtengan los resultados esperados sin la utilización de una parte conocida. Como la inclusión de una parte conocida permite potencialmente obtener mejores resultados iterando el algoritmo con menores tamaños de población, y por lo tanto por un menor tiempo, podrían probarse partes conocidas distintas (por ejemplo, multiplicar por las distintas variables del problema) iterando el algoritmo con un menor tamaño de población, para así tener tiempo de probar distintas PCs. Finalmente, la parte conocida no sólo se limita a variables que están multiplicando al bloque faltante como ocurre en este caso, sino que también se pueden incluir variables (y también constantes) sumando en el numerador o en el denominador de la expresión propuesta por los individuos.

4.1.7. Estudio del efecto del error en los datos experimentales

Para estudiar el efecto de introducir error a los datos experimentales en el desempeño del algoritmo se hicieron tres pruebas, al sumar tres bandas de error distintas a los datos de entrenamiento y de validación previamente simulados: $\pm 0,5$, ± 1 y $\pm 1,5$ (g/L). El objetivo de este estudio fue determinar si el algoritmo es capaz de proponer modelos que cumplan con la propiedad de generalización a pesar de que los datos experimentales contengan errores. Esto representa una aproximación de la aplicabilidad de la metodología propuesta en problemas que se pueden encontrar en la industria de procesos, debido a que los datos experimentales siempre tendrán algún grado de error. En este caso de estudio el error en los datos experimentales puede deberse a que la concentración de células no es homogénea en el reactor, a que la toma de muestra es imprecisa o puede ser propio del instrumento o técnica utilizado para medir la concentración de células.

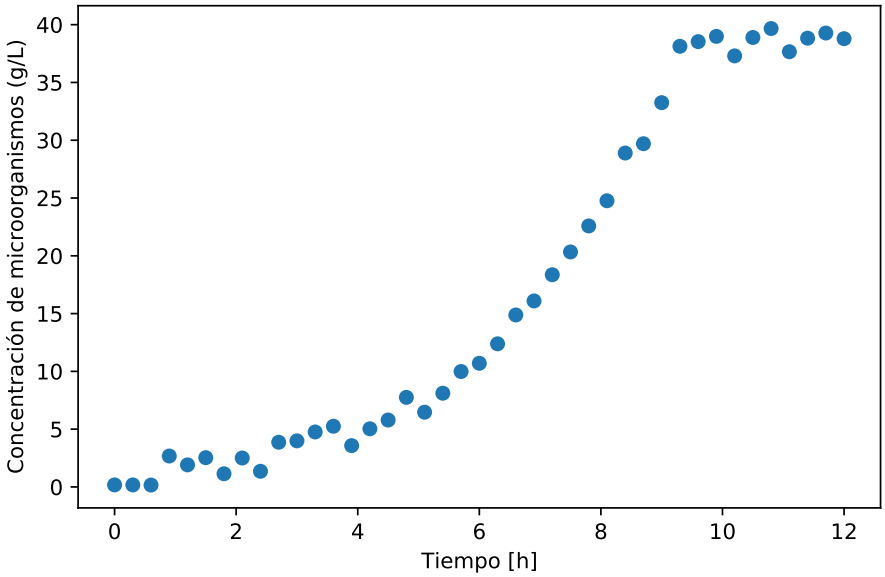


Figura 4.13: Datos experimentales con banda de error de $\pm 1,5$.

Para calcular el fitness del modelo matemático propuesto por cada individuo, la función de evaluación diseñada lo simula utilizando como condiciones iniciales al primer dato experimental disponible de cada variable. Por lo tanto, si los datos experimentales tienen error, las condiciones iniciales utilizadas por el algoritmo también tendrán este error y el modelo se estará simulando bajo condiciones distintas a las que se tenían cuando se generaron los datos. Este problema se acentúa más mientras más sensible es el modelo a la variación de sus condiciones iniciales, siendo el caso más extremo los modelos con dependencia sensible a las condiciones iniciales, también conocidos como modelos caóticos [139]. En este caso de estudio el error en la concentración inicial de microorganismos repercute principalmente en el tiempo que el sistema tarda en llegar al estado estacionario, mientras que el error en la concentración inicial de sustrato afecta mayoritariamente a la concentración de microorganismos en el estado estacionario. En la Figura 4.13 se presentan los datos experimentales con banda de error de $\pm 1,5(g/L)$. Como la concentración de microorganismos se comporta de manera similar a que los datos generados por el modelo original, y la etapa de ejecución de GP sólo tiene como objetivo identificar la estructura matemática del bloque faltante el

cambio de condiciones iniciales no tiene un gran impacto en en esta etapa, pero si lo tiene en la etapa de ajuste de parámetros.

En el gráfico de la Figura 4.14, se muestra la comparación entre simulación del modelo estudiado utilizando las condiciones iniciales originales ($X_0 = 1$ (g/L) y $S_0 = 30$ (g/L)) y las condiciones iniciales con error. Estas nuevas condiciones iniciales corresponden a $X_0 = 0,69$ (g/L) y $S_0 = 29,85$ (g/L) para la banda de $\pm 0,5$ y a $X_0 = 0,18$ (g/L) y $S_0 = 30,23$ (g/L) para la banda de $\pm 1,5$. Se observa que, debido a que en ambos casos de C.I. con error la concentración de microorganismos es menor que la original, el sistema tarda más tiempo en llegar al estado estacionario, lo que se acentúa para el caso de la banda de $\pm 1,5$ que no alcanza a llegar a dicho estado en el tiempo simulado. De todas maneras, en estos casos el error corresponde 82 % del valor original de la concentración inicial de microorganismo, y es poco probable que ocurra en la realidad.

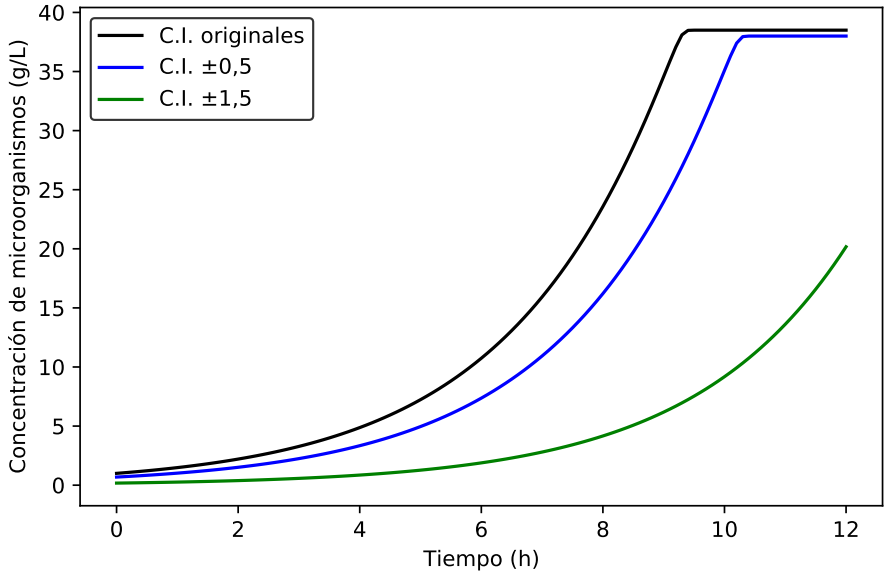


Figura 4.14: Simulación del modelo matemático del caso de estudio a partir de las condiciones iniciales (C.I.) sin error ($X_0 = 1$ y $S_0 = 30$), con banda de error de $\pm 0,5$ ($X_0 = 0,69$ y $S_0 = 29,85$) y con banda de error de $\pm 1,5$ ($X_0 = 0,18$ y $S_0 = 30,23$). Todas las concentraciones medidas en (g/L).

Entonces, si se utilizan las condiciones iniciales con error el modelo debería tener parámetros (μ y K_m) distintos a los originales para que se ajuste a los datos de entrenamiento. En la Tabla 4.5 se presentan los parámetros necesarios para que esto ocurra, y se observa que estos difieren notablemente de los originales ($\mu = 0,4$ y $K_m = 0,3$), sobre todo K_m. Por lo tanto, es necesario contar con las C.I. originales en la etapa de ajuste para obtener los valores de μ y K_m que realmente describen el fenómeno estudiado, ya que en caso contrario se va a generar un caso de sobreajuste a los datos experimentales de entrenamiento en el que el error en las C.I. será compensado por los nuevos parámetros.

Tabla 4.5: Parámetros necesarios para que el modelo se ajuste a los datos de entrenamiento si se utilizan las condiciones iniciales con error.

Banda de error (g/L)	**Condiciones iniciales** (g/L)	**Parámetros**	**Intervalo de confianza (95 %)**
$\pm 0,5$	$X_0 = 0,69$	$\mu = 0,480$	$\pm 0,0184$
	$S_0 = 29,85$	$K_m = 1,95$	$\pm 0,904$
± 1	$X_0 = 0,18$	$\mu = 1,05$	$\pm 0,321$
	$S_0 = 30,23$	$K_m = 17,3$	$\pm 12,7$

Para este caso de estudio se utilizaron las condiciones iniciales con error para la etapa de ejecución de GP y las condiciones iniciales originales en las etapas de ajuste de parámetros y validación. Se realizaron 50 repeticiones del algoritmo, con 50 generaciones de 500 individuos cada una. Para cada banda de error se escogió el modelo matemático que tuvo el mejor desempeño en la etapa de validación. En la Tabla 4.6 se muestran la expresión matemática, fitness ajustado y fitness de validación del mejor individuo para las distintas bandas de error. Además, se muestra el fitness ajustado y de validación entre la simulación del modelo matemático original y los datos experimentales con error (fitness ajustado ref. y fitness validación ref., respectivamente).

Tabla 4.6: Individuos con mejor fitness de validación para las distintas bandas de error. Resultados obtenidos utilizando condiciones iniciales sin error.

Banda de error (g/L)	**Expresión individuo**	**Fitness ajustado**	**Fitness ajustado ref.**	**Fitness validación**	**Fitness validación ref.**
$\pm 0,5$	$\frac{0,736 \cdot S \cdot X^2}{1,84 \cdot S \cdot X + 0,555 \cdot X + 0,0253}$	0,112	0,112	0,0868	0,0877
± 1	$\frac{0,0584 \cdot S + 0,798 \cdot S \cdot X^2}{2 \cdot S \cdot X + 0,926 \cdot X}$	0,347	0,357	0,392	0,358
$\pm 1,5$	$\frac{1,13 \cdot S \cdot X}{-6,55 \cdot 10^{-4} \cdot S \cdot X + 2,84 \cdot S + 0,847}$	0,775	0,729	0,827	0,798

En la Tabla 4.6 se puede apreciar que las expresiones propuestas en los tres casos tienen una estructura matemática similar a la del bloque faltante original, difiriendo sólo en la inclusión de una variable extra en cada uno de ellos, que luego de realizar el test de significancia de los parámetros es eliminada en todos los casos (ver Anexo A.2). En la Tabla 4.7 se muestra el análisis estadístico de los parámetros significativos de cada modelo. Además, se observa que los tres modelos planteados tienen un desempeño cercano al utilizado para generar los datos experimentales, ya que tanto el fitness ajustado como el de validación tienen un valor similar (en casi todos los casos del mismo orden de magnitud) que su valor de referencia. En el gráfico de la Figura 4.15 se presenta la simulación del modelo obtenido para la banda de error de $\pm 1,5$ (g/L) junto con los datos experimentales de entrenamiento y de validación, y se ratifica que dicho modelo se ajusta de buena manera a los datos.

Tabla 4.7: Análisis estadístico de los parámetros de los mejores individuos obtenidos.

Banda de error (g/L)	**Expresión ajustado**	**Fitness validación**	**Fitness**	**Parámetro**	**p-value**	**IC (95 %)**
$\pm 0,5$	$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X}$	$0,105$	$0,196$	$p_1 = 0,743$	$7{,}55 \cdot 10^{-15}$	$\pm 0,122$
				$p_2 = 1,87$	$3{,}47 \cdot 10^{-13}$	$\pm 0,349$
				$p_3 = 0,362$	$0{,}467$	$\pm 0,997$
± 1	$\frac{p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,347$	$0,357$	$p_2 = 0,316$	$0{,}00310$	$\pm 0,202$
				$p_3 = 0,780$	$0{,}00535$	$\pm 0,534$
				$p_4 = 0,432$	$0{,}217$	$\pm 0,697$
$\pm 1,5$	$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_4}$	$0,721$	$0,863$	$p_1 = 0,527$	$9{,}91 \cdot 10^{-7}$	$\pm 0,183$
				$p_2 = 1,31$	$8{,}76 \cdot 10^{-6}$	$\pm 0,517$
				$p_4 = 0,420$	$0{,}621$	$\pm 1,71$

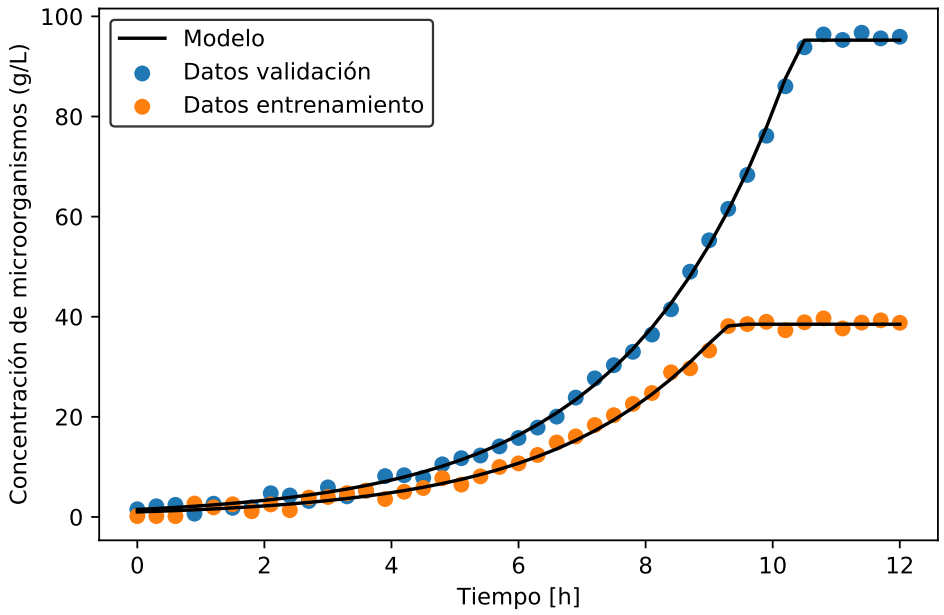


Figura 4.15: Simulación del modelo propuesto por el mejor individuo para la banda de error de $\pm 1,5$ (g/L).

Si no se tiene información sobre las C.I. originales y sólo se cuenta con los datos experimentales con error, las C.I. de dichos datos pueden incluirse dentro de los parámetros por optimizar en la etapa de ajuste de parámetros. En la Tabla 4.8 se presentan los modelos de la Tabla 4.6, pero con los parámetros ajustados utilizando el método anteriormente descrito. La columna fitness validación corresponde al obtenido al validar el modelo utilizando las condiciones iniciales sin error ($X_0 = 1,5$ y $S_0 = 75$), mientras que la columna fitness validación 2 corresponde a la validación de los modelos utilizando como condiciones iniciales el primer dato disponible de cada variable (C.I. con error).

Tabla 4.8: Ajuste de parámetros de los mejores individuos incluyendo las condiciones iniciales dentro de los parámetros optimizados.

Banda de error (g/L)	**Expresión**	**Fitness ajustado**	**Fitness validación**	**Fitness validación 2**	**Parámetro**	**p-value**	**IC (95 %)**
$\pm 0,5$	$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X}$	$0,0931$	2,14	53,4	$X_0 = 0,949$	0	$\pm 0,079$
					$S_0 = 30,12$	0	$\pm 0,187$
					$p_1 = 0,695$	0	$\pm 0,0848$
					$p_2 = 1,72$	$2,86 \cdot 10^{-14}$	$\pm 0,288$
					$p_3 = 0,358$	0,570	$\pm 1,27$
± 1	$\frac{p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,351$	0,925	170	$X_0 = 1,01$	$1,55 \cdot 10^{-14}$	$\pm 0,165$
					$S_0 = 30,05$	0	$\pm 0,364$
					$p_2 = 1,03$	0,00185	$\pm 0,845$
					$p_3 = 2,54$	0,00345	$\pm 2,35$
					$p_4 = 1,40$	0,432	$\pm 35,8$
$\pm 1,5$	$\frac{p_1 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$0,640$	9,23	18	$X_0 = 1,01$	$2,99 \cdot 10^{-10}$	$\pm 0,239$
					$S_0 = 30,16$	0	$\pm 0,521$
					$p_1 = 1,48$	0,352	$\pm 3,18$
					$p_3 = 3,53$	0,389	$\pm 8,22$
					$p_4 = 4,89$	0,0894	$\pm 5,69$

Para las bandas de error $\pm 0,5$ y $\pm 1,5$ se observa que los modelos obtenidos son menos generalizados que los obtenidos previamente (Tabla 4.6) debido a que tienen un mejor fitness ajustado, pero un peor fitness de validación. Mientras que el modelo obtenido para la banda de error ± 1 tiene un peor desempeño en ambas etapas, pero la diferencia es mínima, como se puede observar en el gráfico de la Figura 4.16, donde el Modelo 1 corresponde al obtenido luego de ajustar los parámetros utilizando las C.I. sin error y el Modelo 2 corresponde al caso en que las C.I. fueron incluidas dentro de la optimización de parámetros.

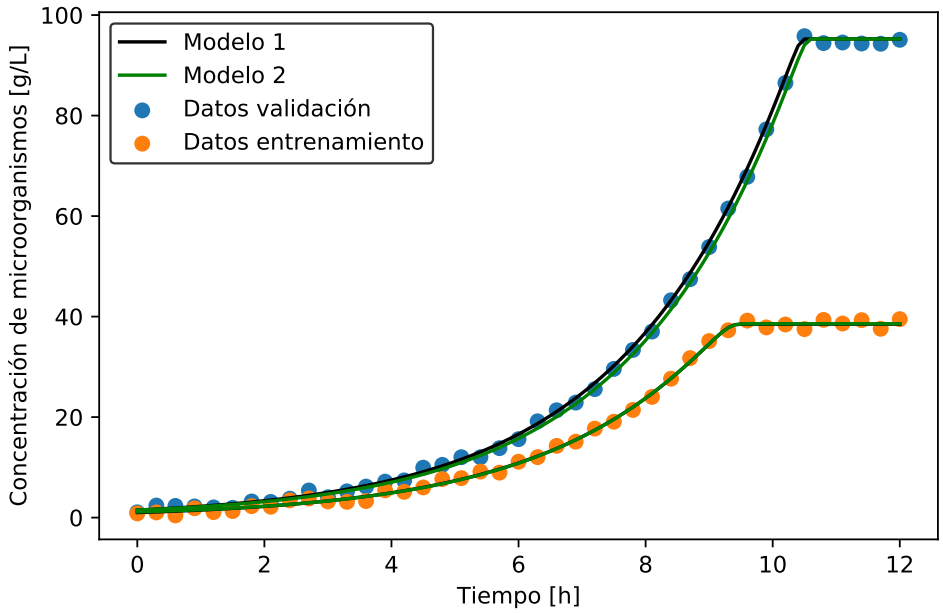


Figura 4.16: Simulación del modelo propuesto por el mejor individuo para la banda de error ± 1 (g/L) luego de ajustar sus parámetros utilizando las C.I. si error (Modelo 1) e incluyendo las C.I. como parámetro en la optimización (Modelo 2).

Entonces, es posible identificar la estructura matemática del bloque faltante e incluso obtener una aproximación de sus parámetros si se cuenta con un conjunto de datos de entrenamiento cuyas C.I. presentan error. Los modelos identificados van a tener una menor capacidad de generalización, lo que se evidencia con la disminución del fitness de validación entre los modelos presentados en la Tabla 4.6 y los de la Tabla 4.8, pero aún así son una mejor alternativa que los modelos obtenidos al utilizar las C.I con error (sin optimizarlas) y pueden utilizarse para tener una aproximación al comportamiento del fenómeno estudiado cuando opera bajo condiciones distintas a las de entrenamiento. Por ejemplo, en el gráfico de la Figura 4.17 muestra que el modelo obtenido luego de optimizar las condiciones iniciales con error de $\pm 1,5$ (g/L) (Modelo 2) no difiere en gran medida de los datos experimentales de validación a pesar de que su fitness de validación es 13. La concentración en estado estacionario no varía con respecto a la del Modelo 1, y la mayor diferencia es que se llega más rápido a dicho estado. Para mejorar la generalización de la etapa de ajuste de parámetros se podrían aplicar técnicas de remuestreo como Jackknife o bootstrapping [140].

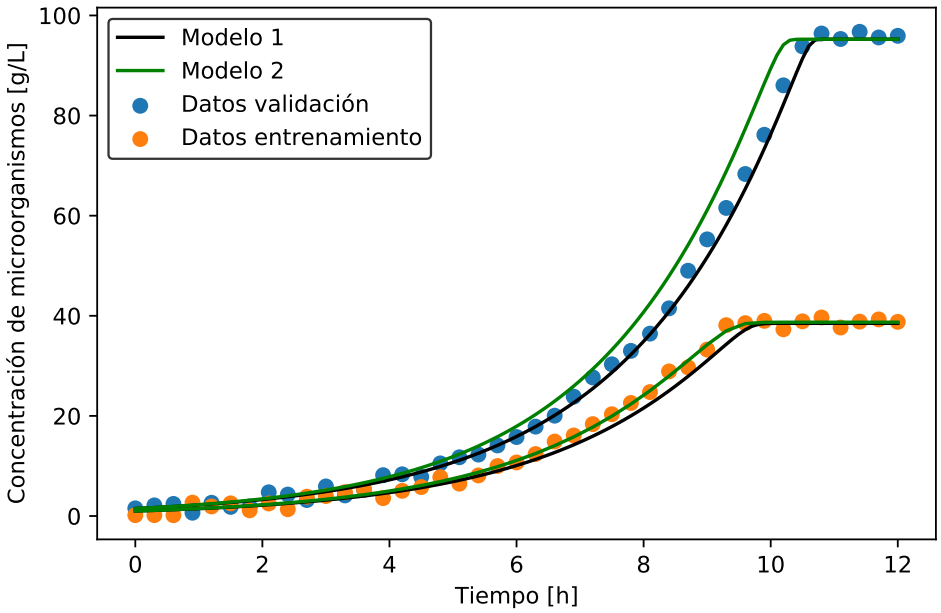


Figura 4.17: Simulación del modelo propuesto por el mejor individuo para la banda de error $\pm 1,5$ (g/L) luego de ajustar sus parámetros utilizando las C.I. sin error (Modelo 1) e incluyendo las C.I. como parámetro en la optimización (Modelo 2).

Pese a que se pueden obtener resultados utilizando las C.I. con error en el conjunto de datos de entrenamiento, es fundamental que las C.I. del conjunto de datos de validación no tengan error para poder evaluar con certeza el desempeño de los modelos generados. En la Tabla 4.8 se evidencia que el fitness de validación puede variar hasta en tres órdenes de magnitud entre los casos de C.I. con y sin error.

4.2. Caso de estudio 2: Producción simultánea de etanol y fructosa a partir de sacarosa

El segundo caso de estudio también corresponde a un caso sintético, debido a que los datos experimentales se obtuvieron simulando un modelo matemático. Sin embargo, este caso tiene por objetivo simular la aplicación de la metodología propuesta en un problema de mayor complejidad que se puede encontrar en la industria de procesos. Los parámetros y condiciones iniciales utilizados para la obtención de los datos se basan en el estudio realizado por Sulieman y col. (2018) sobre la producción simultánea de etanol y fructosa a partir de sacarosa [46]. Esto se logra mediante la fermentación selectiva de glucosa por sobre la fructosa para sintetizar bioetanol con levaduras (*Saccharomyces cerevisiae* ATCC 36858).

Se seleccionó este caso de estudio debido a que corresponde a un ejemplo de producción de más de un compuesto utilizando microorganismos, por lo que puede ser representativo de la aplicabilidad de la metodología diseñada para la identificación de funciones en sistemas diferenciales utilizados en la industria de procesos biotecnológicos para la producción de otros tipos de compuestos. Además, como se muestra en la Sección 4.2.1, este caso de estudio corresponde a un modelo matemático relativamente complejo, compuesto por 5 ecuaciones diferenciales y 10 parámetros distintos, por lo que también es de utilidad para identificar debilidades y fortalezas de la metodología en modelos matemáticos que tengan muchas variables y parámetros.

4.2.1. Modelo matemático

Como se presentó en la Sección 2.2.2, el modelo matemático que describe la producción de etanol con levaduras a partir de sacarosa está compuesto por las Ecuaciones 4.10-4.14, que representan la evolución en el tiempo de las concentraciones de levaduras, sacarosa, glucosa, fructosa y etanol, respectivamente.

$$\frac{dX}{dt} = \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E_m}) \cdot X \tag{4.10}$$

$$\frac{dS}{dt} = -\alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \tag{4.11}$$

$$\frac{dG}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{G/S} - \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E'_m}) \cdot X \cdot \frac{1}{Y_{X/G}} \tag{4.12}$$

$$\frac{dF}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{G/S} - \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E'_m}) \cdot X \cdot \frac{R}{Y_{X/G}} \tag{4.13}$$

$$\frac{dE}{dt} = \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E'_m}) \cdot X \cdot \frac{Y_{E/G}}{Y_{X/G}} \cdot (R + 1) \tag{4.14}$$

4.2.2. Datos experimentales

Los parámetros utilizados para la simulación de los datos experimentales (Tabla 4.9) son los propuestos en el estudio realizado por Sulieman y col. (2018)[46]. Además, se usaron condiciones iniciales similares a las de dicho estudio (ver Tabla 4.10). El valor de la proporción entre la tasa de consumo de fructosa y la tasa de consumo de glucosa (R) refleja que la fermentación es selectiva, privilegiando la utilización de glucosa por sobre la fructosa. La concentración inicial de sacarosa es menor que la reflejan que la sacarosa fue sometida a un pretratamiento, siendo hidrolizada en glucosa y fructosa.

Tabla 4.9: Parámetros utilizados para la obtención de los datos experimentales en el caso de estudio 2 [46].

Parámetro	**Valor**	**Parámetro**	**Valor**
μ_{max} (h^{-1})	0,050	K_G (g/L)	0,005
E_m (g/L)	18	α_s (h^{-1})	0,024
K_S (g/L)	0,003	$Y_{G/S}$ (g/g)	2,24
E'_m (g/L)	53,6	$Y_{X/G}$ (g/g)	0,070
R (g/g)	0,165	$Y_{E/G}$ (g/g)	0,434

Tabla 4.10: Condiciones iniciales utilizadas para la simulación de los datos experimentales de entrenamiento para el caso de estudio 2 [46].

Especie	X	S	G	F	E
Concentración inicial (g/L)	1	10	65	65	0

Como se muestra en los gráficos de la Figura 4.18, el tiempo total simulado fue de 168 horas (7 días) para obtener información de dinámica del sistema y una aproximación de su estado estacionario (las concentraciones siguen variando lentamente). La concentración de glucosa disminuye en el tiempo, mientras que la concentración de fructosa baja un poco en las primeras 36 horas y luego se mantiene prácticamente constante. Este comportamiento se debe a la ya mencionada fermentación selectiva de glucosa por sobre fructosa por parte de las levaduras. Además, la concentración del microorganismo aumenta por las primeras 36 horas aproximadamente, y luego de este punto comienza a decaer porque la concentración de etanol alcanza los 18 (g/L), que es la concentración a partir de la que se inhibe el crecimiento de las levaduras y la velocidad de muerte celular supera a la de crecimiento (E_m).

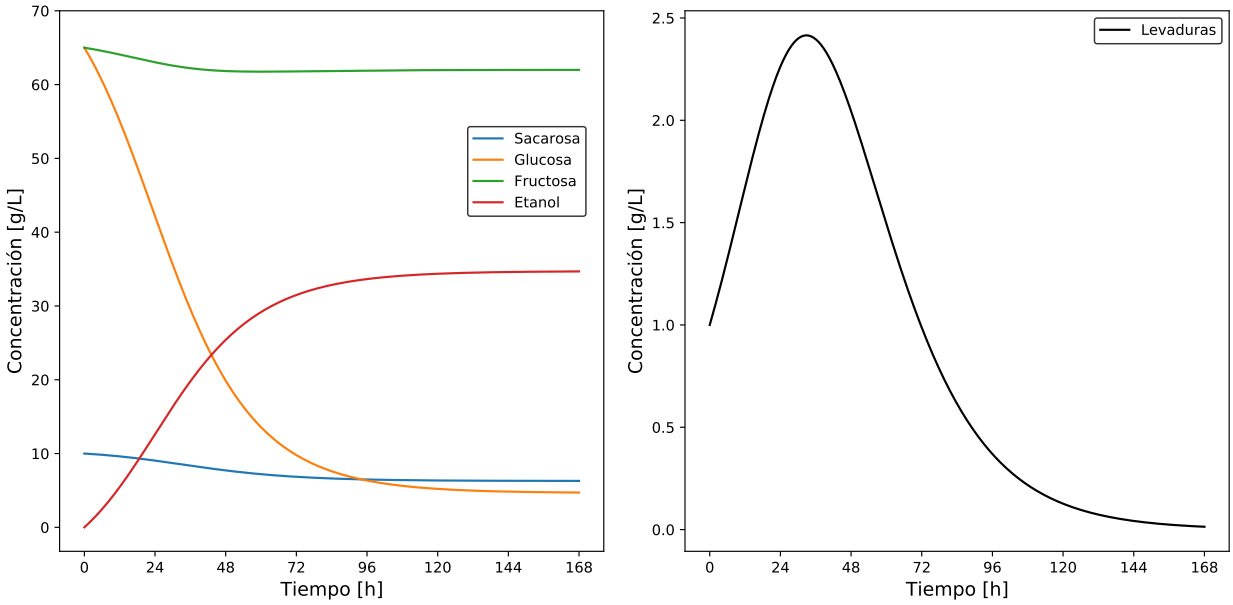


Figura 4.18: Simulación del modelo matemático utilizando las C.I. definidas para la obtención de los datos de entrenamiento.

En la Tabla 4.11 se presentan las condiciones iniciales utilizadas para obtener los datos experimentales de validación y en la Figura 4.19 se muestra la simulación del modelo matemático bajo dichas condiciones. Se observa que la concentración de glucosa desciende rápidamente hasta llegar a 0 (g/L), y en ese punto se produce un quiebre en las velocidades de consumo y producción de todas las especies (menos sacarosa) porque estas velocidades dependen de la concentración de glucosa. Luego de ese primer quiebre, la concentración de glucosa se mantiene en 0 (g/L) porque se consume a medida que se va produciendo por la hidrólisis de sacarosa. Por lo tanto, el objetivo del conjunto de condiciones iniciales de validación utilizadas es verificar si los modelos planteados por el algoritmo GP son capaces de describir el comportamiento del sistema en casos en que la glucosa se agota rápidamente.

Tabla 4.11: Condiciones iniciales utilizadas para la simulación de los datos experimentales de validación para el caso de estudio 2.

Especie	X	S	G	F	E
Concentración inicial (g/L)	4	15	20	20	0

Como ya se observó en la en la Figura 4.18, la concentración de levaduras va en aumento hasta que la concentración de etanol alcanza los 18 (g/L) (a las 55 horas aprox.), luego de eso comienza a disminuir. Otro aspecto importante que se puede observar en la Figura 4.19 es la evolución de la concentración de fructosa. Debido a que su velocidad consumo depende de la concentración de glucosa, su concentración disminuye hasta que la glucosa llega a 0 (g/L) y luego comienza a aumentar debido a que se genera (por la hidrólisis de sacarosa) a mayor velocidad de la que se consume. El estado estacionario se alcanza cuando se acaba la sacarosa, por lo que no se puede producir más glucosa ni fructosa, y como en este modelo representa una situación en la que no se consume fructosa sin la presencia de glucosa, las

concentraciones de etanol y fructosa se mantienen constantes. La concentración de levaduras también se mantiene constante cuando se acaba la glucosa, debido a que el modelo no incluye un término independiente de esta especie para la muerte celular.

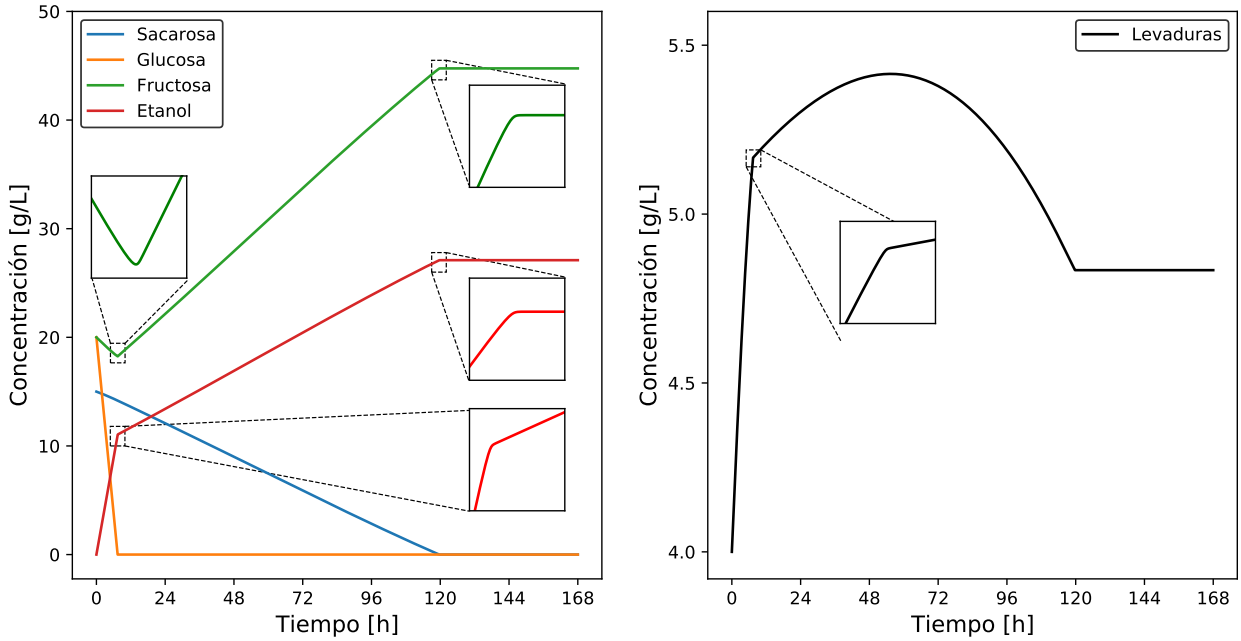


Figura 4.19: Simulación del modelo matemático utilizando las C.I. definidas para la obtención de los datos de validación.

El modelo matemático propuesto por Sulieman y col. (2018) tiene dos debilidades. La primera es que la tasa de consumo de la fructosa depende directamente de la concentración de glucosa (Ecuación 4.13), mientras más glucosa hay se consume más fructosa. El objetivo de la fermentación selectiva es privilegiar el consumo de glucosa por sobre el de fructosa, por lo que no debería usarse fructosa cuando se tiene alta concentración de glucosa. Si llegase a haber consumo de fructosa debería darse cuando haya menor disponibilidad de glucosa, pero como la cepa ATCC 36858 no tiene hexoquinasas tampoco se puede metabolizar fructosa en estas condiciones [141] [142]. Esta última situación si es bien representada por el modelo. La segunda debilidad, y la más importante, es la Ecuación 4.10, que describe la tasa de crecimiento o muerte celular. El problema con esta ecuación es que depende totalmente de la concentración de glucosa, al punto en que si se acaba esta fuente de carbono los microorganismos dejan de morir, lo que se aleja de la realidad porque es justamente la falta de sustrato la que hace que las levaduras mueran (además de una alta concentración de etanol). Una solución sería sumar un término que contenga una tasa específica de muerte celular independiente de la concentración.

Para calcular el fitness de los individuos se utilizaron los datos experimentales de la concentración de etanol porque este compuesto corresponde al principal producto de interés. Se simuló que la toma de muestras se realiza cada cuatro horas, lo que se aleja de la realidad debido a que es poco probable que se pueda tomar muestras en horarios no laborales, a menos que se cuente con sensores que estén continuamente midiendo la concentración de etanol dentro del fermentador, como por ejemplo biosensor [143], un sensor químico [144] o

un sensor basado en el análisis infrarrojo de gases [145]. Además, se generó una banda de error de $\pm 0,5$ (g/L) a los datos simulados, de manera de acercar este problema sintético a un caso de la industria de procesos. En la Figura 4.20 se presentan los datos experimentales de entrenamiento y de validación antes mencionados.

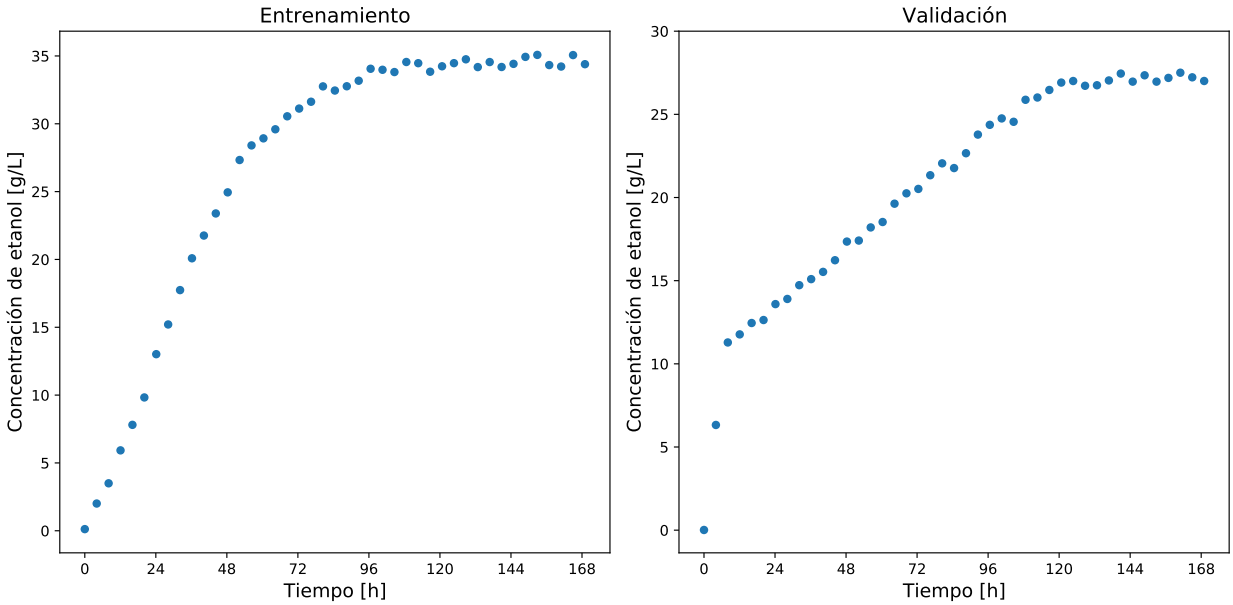


Figura 4.20: Datos experimentales de entrenamiento y de validación utilizados en el caso de estudio 2.

4.2.3. Bloque faltante estudiado

Debido a que el principal producto de interés es el etanol, en este caso de estudio se considera como expresión desconocida la tasa específica de producción de etanol (q_E), por lo que la expresión del bloque faltante es la que se muestra en la Ecuación 4.15.

$$BF = q_E = 0,05 \cdot (\frac{G}{0,005 + G}) \cdot (1 - \frac{E}{53,6}) = \frac{2,68 \cdot G - 0,05 \cdot E \cdot G}{53,6 \cdot G + 0,268} \tag{4.15}$$

Por lo tanto, para ejecutar el algoritmo GP, se deben incluir como ecuaciones conocidas a las Ecuaciones 4.16-4.20, donde EI corresponde a la expresión propuesta por cada individuo para el bloque faltante. Nótese que la Ecuación 4.19 puede ser prescindible, ya que la concentración de fructosa no es parte de ninguna ecuación diferencial por lo que no influye en las demás variables del sistema. Por lo tanto, esta ecuación podría no incluirse en caso de tener el conocimiento previo de que la variable F forma parte del bloque que se pretende identificar.

$$\frac{dX}{dt} = 0,05 \cdot (\frac{G}{0,005 + G}) \cdot (1 - 0,05556 \cdot E) \cdot X \tag{4.16}$$

$$\frac{dS}{dt} = -0,024 \cdot (\frac{S}{0,003 + S}) \cdot X \tag{4.17}$$

$$\frac{dG}{dt} = 0,05376 \cdot (\frac{S}{0,003 + S}) \cdot X - 14,29 \cdot EI \cdot X \tag{4.18}$$

$$\frac{dF}{dt} = 0,05376 \cdot (\frac{S}{0,003 + S}) \cdot X - 2,357 \cdot EI \cdot X \tag{4.19}$$

$$\frac{dE}{dt} = 7,223 \cdot EI \cdot X \tag{4.20}$$

El objetivo de este caso de estudio es simular la aplicación de la metodología planteada en un problema más complejo en el que no se conoce el bloque faltante. Por lo tanto, en los resultados presentados a continuación no se realiza una comparación entre los modelos obtenidos y la estructura y parámetros de la función original como se hizo en el caso de estudio 1. Todas las decisiones se toman en función del desempeño de los modelos obtenidos en la etapa de validación.

4.2.4. Primera iteración

En la primera iteración se realizaron 20 repeticiones del algoritmo GP con 500 individuos y 50 generaciones. Posteriormente, siguiendo la metodología propuesta, se ajustaron los parámetros del mejor individuo de cada repetición y se validó el modelo obtenido. En la Tabla 4.12 se presenta el individuo que tuvo mejor desempeño en la etapa de validación y se evidencia que la expresión matemática propuesta para la tasa específica de producción de etanol (q_E) tiene una estructura similar al modelo de cinética enzimática de Michaelis-Menten, con la expresión $\cdot G \cdot X$ en el denominador actuando como una especie de inhibición. Además, esta expresión no incluye términos de inhibición por presencia de etanol. El fitness ajustado es 0,138 y el fitness de validación es 7,19.

Tabla 4.12: Individuo con mejor fitness de validación luego ejecutar el algoritmo GP por 20 repeticiones por 50 generaciones de 500 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G}{p_2 \cdot G + p_3 \cdot G \cdot X + p_4}$	$0,138$	7,19	$p_1 = 2,78$	0,0150	$\pm 2,23$
			$p_2 = 13,5$	0,0381	$\pm 12,8$
			$p_3 = 18,0$	0,0101	$\pm 13,6$
			$p_4 = 985$	0,0133	± 776

En los gráficos de la Figura 4.21 se muestra la simulación de la evolución de concentración de etanol utilizando las C.I. de entrenamiento y de validación. Se aprecia que corresponde a un modelo sobreajustado ya que se describe de buena manera a los datos de entrenamiento, pero en la validación tiene un comportamiento de curva de saturación que no representa fielmente

a los datos experimentales, ya que la síntesis de etanol es más lenta antes del primer quiebre (a las 12 horas aprox.), lo que puede deberse a la expresión $\cdot G \cdot X$ actuando como inhibidor o al alto valor de la constante libre en el denominador, que sería un equivalente a K_G en la cinética de Michaelis-Menten. Luego del primer quiebre, la formación de etanol del modelo es más rápida que la de los datos, llegando unas 45 horas antes al estado estacionario. La concentración final es igual que la de los datos experimentales.

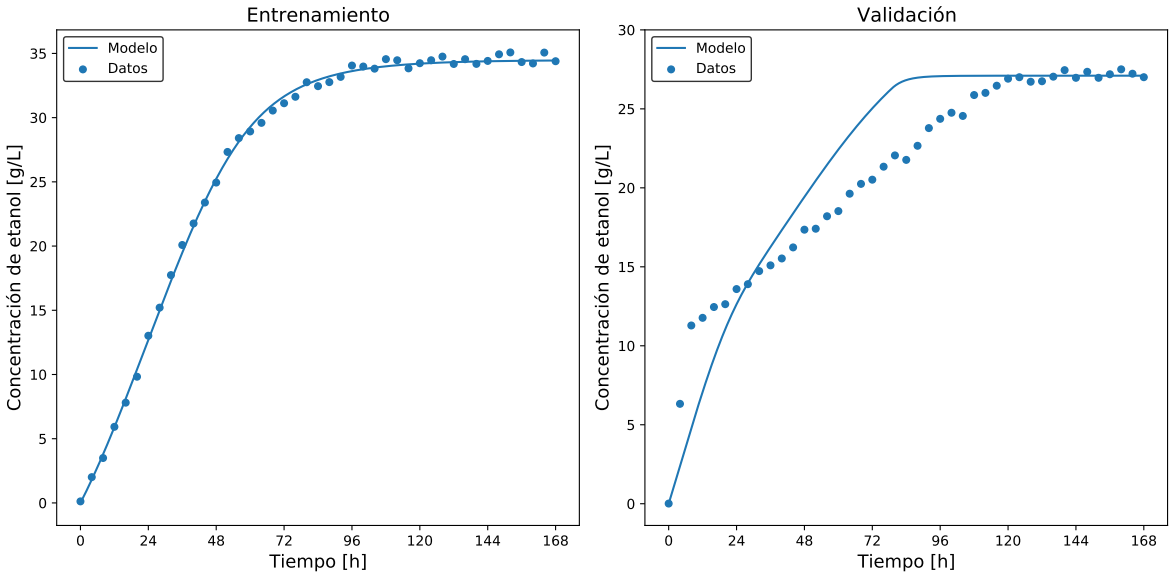


Figura 4.21: Evolución de la concentración de etanol en el tiempo. Simulado a partir del mejor modelo obtenido en la primera iteración.

Para comprender el motivo del comportamiento de la concentración de etanol en el modelo planteado es necesario conocer la evolución de la concentración de las demás especies presentes en el reactor. En los gráficos de la Figura 4.22 se presenta la simulación del modelo obtenido con las condiciones iniciales de los datos validación. En los datos experimentales la concentración de glucosa desciende rápidamente hasta llegar a 0 (g/L), y la fructosa baja su concentración hasta que se acaba la glucosa, luego de eso comienza a subir hasta que se acaba la sacarosa, momento en el que también se deja de producir etanol. Este comportamiento sugiere que la tasa específica de producción de etanol (bloque faltante buscado) depende de la concentración de glucosa. En el modelo planteado, la tasa de producción de etanol es más lenta en las primeras horas, por lo que la fructosa es consumida a una velocidad menor de la que es producida y comienza a acumularse desde el principio. La glucosa también se consume más lento que en los datos experimentales y su concentración parece estancarse entre las 24 y 75 horas, luego de eso deja de producirse porque se acaba la sacarosa y se consume hasta agotarse.

La mayor disponibilidad de glucosa hace que la concentración de levaduras aumente a casi el doble del valor de los datos experimentales. Esta mayor concentración de microorganismos es lo que genera que la sacarosa se hidrolice con más velocidad que en los datos experimentales, contribuyendo al aumento de concentración de fructosa y al estancamiento de la concentración de glucosa entre las 24 y 75 horas. Al alcanzar 18 (g/L) de etanol la concentración de levaduras va a comenzar a disminuir independiente de cuál sea la expresión del bloque faltante, ya

que dicha expresión no afecta directamente a la tasa de crecimiento del microorganismo, como se muestra en la Ecuación 4.16. Esta ecuación se hace negativa cuando se llega a dicha concentración umbral y la velocidad muerte celular es mayor si hay más glucosa y microorganismos, lo que explica la menor concentración final de levaduras en el modelo planteado.

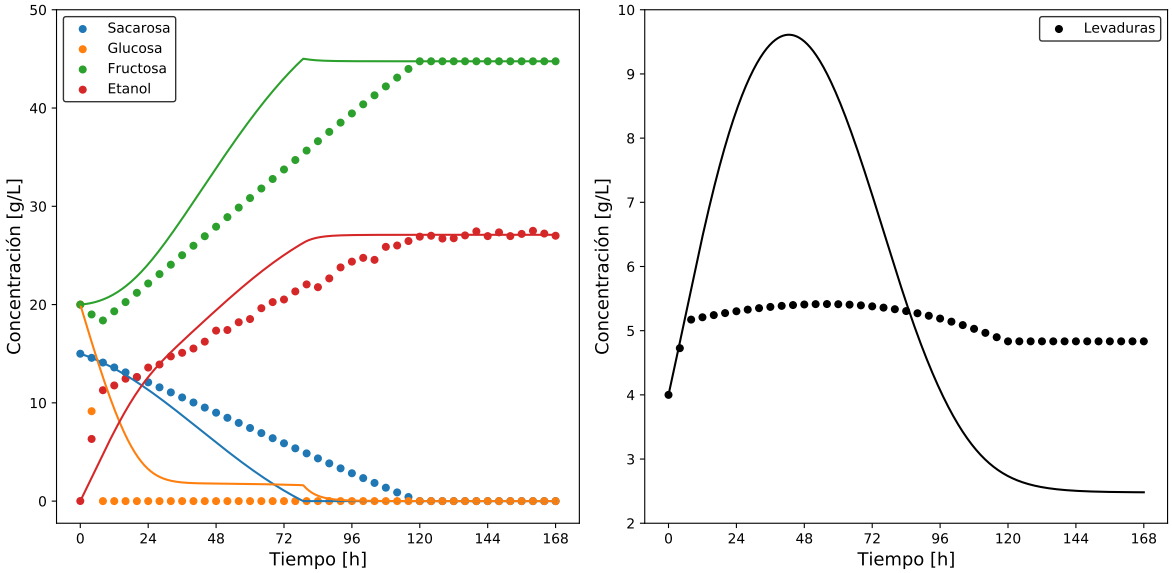


Figura 4.22: Simulación del modelo propuesto por el mejor individuo de la primera iteración con las C.I. de validación.

Entonces, el motivo de la mayor velocidad de síntesis de etanol en el modelo con respecto a los datos experimentales es que hay una mayor concentración de glucosa y de microorganismos. La concentración de glucosa tiene efecto principalmente en la tasa de formación específica de etanol (q_E), que corresponde al bloque faltante, y la concentración de microorganismos afecta porque multiplica a esta función, como se muestra en la Ecuación 4.20.

4.2.5. Segunda iteración

La segunda iteración tiene como objetivo identificar un modelo matemático que tenga una mejor generalización, es decir, que tenga un fitness de validación menor que el modelo propuesto en la primera iteración. Para lograrlo, se siguió la recomendación propuesta en la Sección 4.1.5 de privilegiar la utilización de una población de mayor tamaño en lugar de aumentar el número de generación. Por lo tanto, al igual que en la primera iteración, se realizaron 20 repeticiones, cada una de ellas 50 generaciones, pero se duplicó el tamaño de población de 500 a 1000 individuos por generación. De acuerdo con la metodología diseñada, se ajustaron los parámetros del mejor individuo de cada repetición y se validó el modelo obtenido. En la Tabla 4.13 se presenta el individuo que tuvo mejor desempeño en la etapa de validación. Como ya se mencionó, este modelo se acerca más a los datos experimentales de validación que el obtenido previamente, pero aún no logra describir fielmente la evolución de la concentración de las distintas especies presentes.

Tabla 4.13: Individuo con mejor fitness de validación luego ejecutar el algoritmo GP por 20 repeticiones por 50 generaciones de 1000 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G + p_2 \cdot G^2}{p_3 \cdot G + p_4 \cdot X}$	$0,0949$	5,75	$p_1 = 2,17$	0,0421	$\pm 2,09$
			$p_2 = 0,0698$	0,0488	$\pm 0,00694$
			$p_3 = 128$	0,0474	± 126
			$p_4 = 96,0$	0,0474	$\pm 9,48$

La expresión obtenida es más compleja que la de la primera iteración, ya que tiene más variables al incorporar las expresiones $\cdot G^2$ en el numerador y $\cdot G \cdot S$ en el denominador. Este modelo obtenido también corresponde a un caso de sobreajuste porque tiene un buen fitness ajustado, pero no tiene tan buen desempeño en la etapa de validación. De todas maneras su fitness de validación es menor que el obtenido en la primera iteración, por lo que el aumento de individuos representa una mejora. Sin embargo, esta mejora no es suficiente para lograr identificar un modelo generalizado del fenómeno estudiado. En la Figura 4.23 se muestra la evolución de la concentración de etanol obtenida al simular el modelo propuesto por el mejor individuo y se puede apreciar que tiene un comportamiento de curva de saturación similar al observado en la validación del mejor individuo de la primera iteración, pero en este caso la producción de etanol es más rápida en las primeras 24 horas y más lenta luego de ese punto, acercándose más a los datos experimentales. El aumento en la tasa de producción de etanol al principio de la simulación se debe a la inclusión de la variable $\cdot G^2$ con un parámetro positivo en el numerador del bloque faltante. Esto hace que la función propuesta para q_E sea mayor en presencia de glucosa y sacarosa.

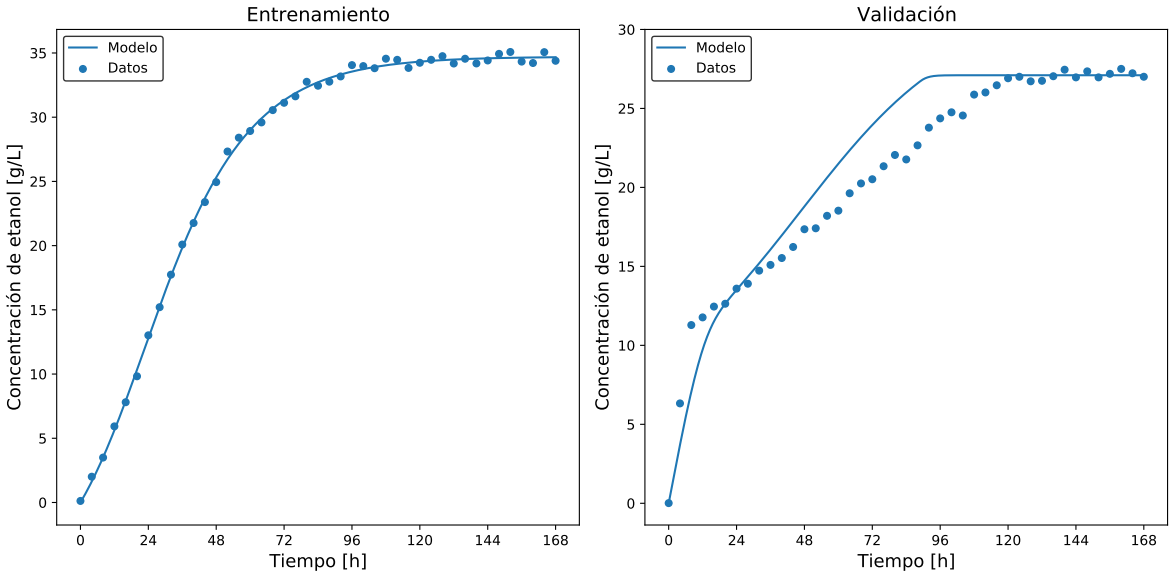


Figura 4.23: Evolución de la concentración de etanol en el tiempo. Simulado a partir del mejor modelo obtenido en la segunda iteración.

En los gráficos de la Figura 4.24 se presenta el comportamiento de todas las especies del sistema al simular el modelo obtenido con las condiciones iniciales de los datos validación. En comparación con la simulación de la primera iteración (Figura 4.22), la glucosa se consume más rápido en las primeras 24 horas, por lo que la concentración de levaduras aumenta menos y la hidrólisis de sacarosa es más lenta. La producción de etanol es más lenta luego de las primeras 24 horas debido a que hay menos glucosa y menos microorganismos.

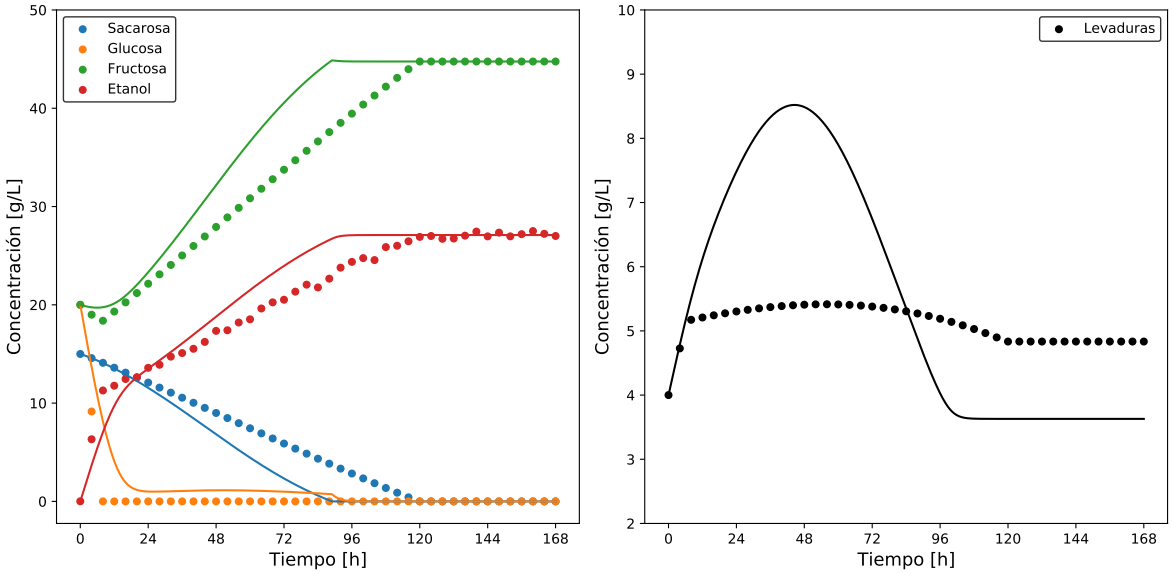


Figura 4.24: Simulación del modelo propuesto por el mejor individuo de la segunda iteración con las C.I. de validación.

Como se mencionó anteriormente, este modelo se acerca más a los datos de validación que el obtenido en la primera iteración, pero aún no representa fielmente el comportamiento del fenómeno estudiado. Para lograr obtener un modelo más generalizado se pueden probar distintas opciones. La primera corresponde a ejecutar el algoritmo con las condiciones de la primera iteración y/o de la segunda iteración por un mayor número de repeticiones, esperando que se encuentre una estructura de la expresión del bloque faltante que se ajuste mejor a los datos experimentales. Otra opción es volver a aumentar el tamaño de población, pero como se muestra en el gráfico de la Figura 4.25, la mediana de los tiempos de ejecución con 500 individuos es de 147 minutos y la de 1000 individuos es de 277 minutos, por lo que es esperable que si se aumenta la población a 1500 el tiempo de ejecución aumentará en aproximadamente dos horas por repetición.

Además, como se discutió en la Sección 4.1.5, un aumento en el número de generaciones no es recomendable en comparación con el aumento en el tamaño de población. Otras alternativas corresponden a la utilización de partes conocidas, que ya se evidenció en la Sección 4.1.6 permiten aumentar significativamente la probabilidad de identificar un modelo generalizado requiriendo tamaños de población más pequeños, por lo que se tarda menos tiempo. Un buen candidato a utilizarse como parte conocida es la expresión $\cdot G$, pues ya se ha identificado que la concentración de glucosa es el principal factor que afecta a q_E. También, como ya se sugirió en la Sección 4.1.6, pueden probarse distintas expresiones "aleatorias" para la parte conocida para un menor tamaño de población y menos repeticiones para cada caso. Pese

a tener todas las opciones expuestas, se decidió estudiar el aporte de la penalización por ausencia de variable (presentada en la Figura 3.4) en la etapa de ejecución del algoritmo GP.

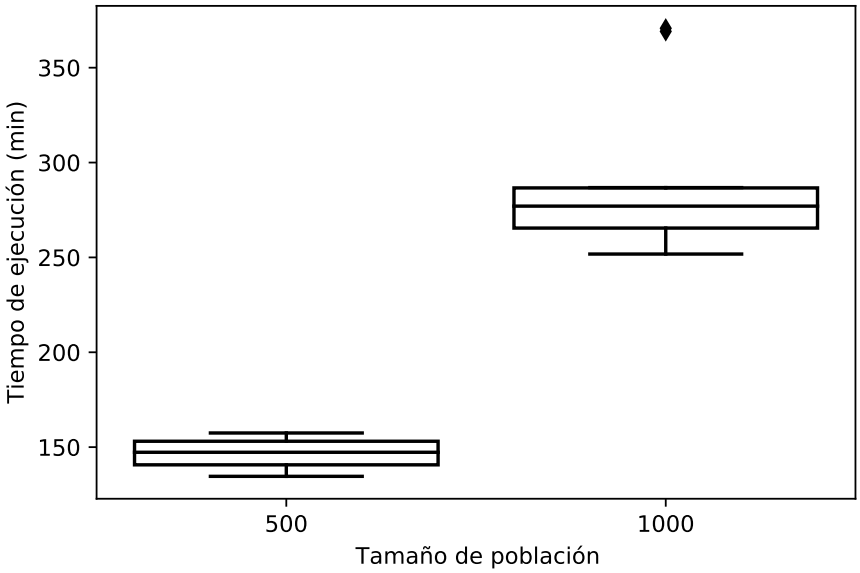


Figura 4.25: Tiempo de ejecución del algoritmo GP para los tamaños de población utilizados en la primera iteración (500) y en la segunda iteración (1000).

4.2.6. Penalización por ausencia de variable

En este caso de estudio se decidió utilizar el conocimiento previo que se tiene sobre la fermentación de etanol a partir de levaduras para guiar la búsqueda del algoritmo GP. Para hacerlo, se decidió penalizar el fitness de los individuos que no contengan la variable de concentración de etanol, basándose en que se sabe que este compuesto inhibe no sólo el crecimiento de la levadura (lo que ya se conocía en la Ecuación 4.16) sino que también es el principal factor de inhibición de la síntesis de dicho producto [146]. En los resultados obtenidos previamente la variable E no forma parte de la expresión identificada para el bloque faltante, lo que podría explicar la falta de generalización de esos modelos.

Cómo se explicó en la Sección 3.0.1.1 (Figura 3.4), la penalización por ausencia de variable funciona buscando dicha variable, en este caso E, en la expresión matemática de cada individuo y si esta variable no se encuentra el individuo es penalizado. Esta penalización puede ser parcial (PP) o total (PT). Si es parcial se calcula el fitness del individuo y luego se multiplica por un factor de penalización, que en este caso es 100. Este valor fue escogido luego de probar distintas constantes: 10, 100, 1.000 y 10.000, realizado 5 repeticiones con cada uno de ellos y obteniendo los mejores resultados con un factor de penalización de 100 (resultados no mostrados en el presente informe). Por otro lado, si es total se asigna un valor arbitrario al fitness del individuo sin realizar ningún cálculo, en este caso el valor asignado fue de 10^{15}.

Para estudiar el efecto de la penalización se realizaron 20 repeticiones de 50 generaciones con 500 individuos para cada tipo de penalización. En la Tabla 4.14 se presenta el mejor modelo obtenido en cada caso junto con el obtenido previamente en la primera iteración (sin penalización). La expresión matemática obtenida con PT tiene una estructura parecida a la expresión identificada con PP, pero es más compleja porque incluye más variables en el numerador ($E \cdot G^2$) y en el denominador tiene $E \cdot G$ en lugar de G. Además, a ambas expresiones les faltan los parámetros que fueron eliminados por no ser singnificantes (ver Anexo B.3).

Tabla 4.14: Mejor individuo obtenido para el caso sin penalizar y en los casos en que se penalizó parcial y totalmente la ausencia de la variable E.

Tipo de penalización	**Expresión**	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
Sin	$\frac{p_1 \cdot G}{p_2 \cdot G + p_3 \cdot G \cdot X + p_4}$	$0,138$	7,19	$p_1 = 2,78$	0,0150	$\pm 2,23$
				$p_2 = 13,5$	0,0381	$\pm 12,8$
				$p_3 = 18,0$	0,0101	$\pm 13,6$
				$p_4 = 985$	0,0133	± 776
Parcial	$\frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_4 \cdot G + p_5}$	$0,102$	0,0649	$p_1 = 134$	$2,53 \cdot 10^{-9}$	$\pm 4,12$
				$p_2 = -2,53$	2	$\pm 0,758$
				$p_4 = 2651$	$1,95 \cdot 10^{-9}$	± 810
				$p_5 = 16,4$	0,986	± 1895
Total	$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G}{p_6 \cdot E \cdot G + p_7}$	$0,0938$	2,22	$p_1 = 26,3$	0,626	± 107
				$p_3 = 3,87$	0,070	$\pm 4,19$
				$p_4 = 115$	0,090	± 133
				$p_6 = 7357$	0,0739	± 8078
				$p_7 = 0,252$	0,999	± 290

La penalización es de utilidad para mejorar la generalización del modelo identificado. El caso más notorio es el de la penalización parcial en que la expresión propuesta por el mejor individuo tuvo incluso mejor desempeño en la validación que en el entrenamiento. La penalización total tiene un mejor fitness de validación que los resultados obtenidos en las primeras dos iteraciones, pero este sigue siendo relativamente alto comparado con el fitness de entrenamiento (cerca de dos órdenes de magnitud de diferencia).

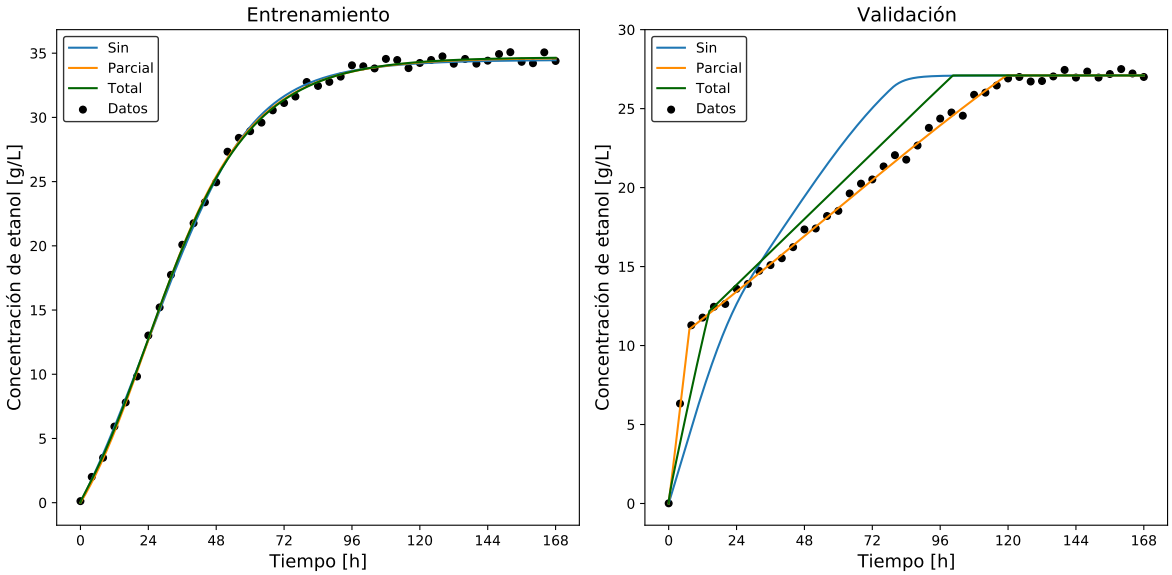


Figura 4.26: Concentración de etanol obtenida al simular los modelos obtenidos sin penalización, con penalización parcial y con penalización total.

En los gráficos de la Figura 4.26 se presenta la comparación de las concentraciones de etanol obtenidas al simular los tres modelos presentados en esta sección. Se observa que todos los modelos tienen un comportamiento prácticamente idéntico al simularse con las C.I. de entrenamiento. En la etapa de validación se aprecia que con la penalización parcial se logra un comportamiento prácticamente idéntico que el de los datos experimentales. El modelo identificado con penalización total tiene una forma similar a los datos experimentales, principalmente porque también presenta un cambio de pendiente muy marcado en las primeras horas de simulación y la segunda pendiente es constante hasta que se llega al estado estacionario. La diferencia radica en que la velocidad de formación de etanol del modelo PT es menor antes del cambio de pendiente y mayor luego de ese punto.

Los gráficos de la Figura 4.27 corresponden a la simulación con las condiciones iniciales de los datos de validación del modelo obtenido con PP. Las concentraciones de sacarosa, glucosa y fructosa parecen ser idénticas que las de los datos experimentales, pero la concentración máxima de levaduras es un poco mayor, lo que sugiere que hay pequeñas diferencias en la concentración de glucosa o de etanol antes de las 24 horas que hacen que la tasa de crecimiento del microorganismo aumente (Ecuación 4.16). Luego de eso, dicha tasa sigue creciendo por la mayor concentración de levaduras, que es la que también hace que su muerte sea más rápida y la concentración final sea menor, debido a que la Ecuación 4.16 depende de X.

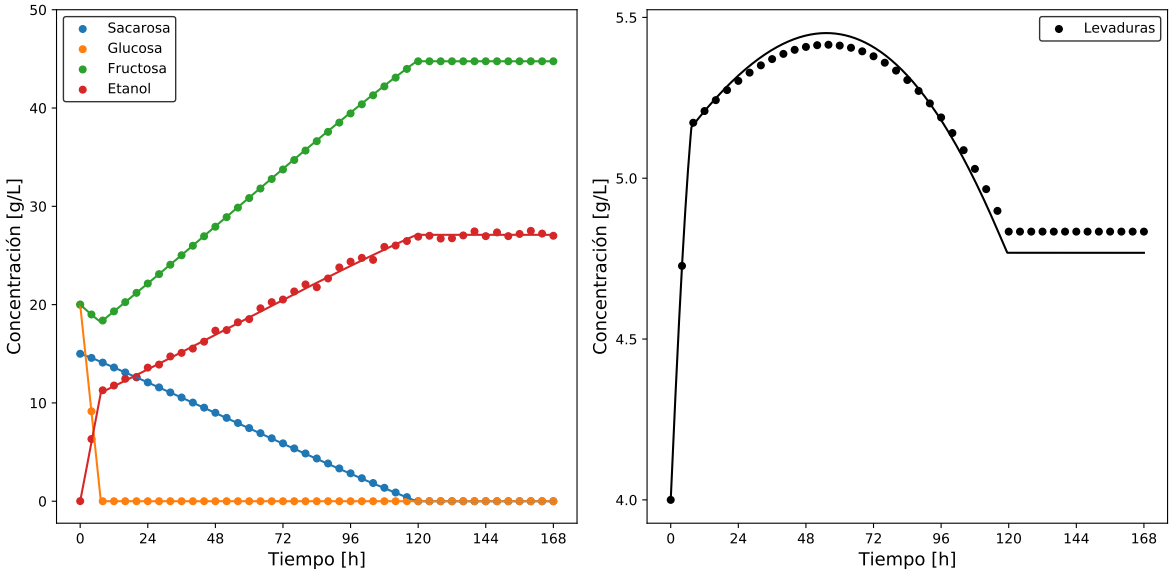


Figura 4.27: Simulación con las C.I. de validación del modelo del mejor individuo obtenido con penalización parcial.

La Figura 4.28 presenta la simulación con las condiciones iniciales de los datos de validación del modelo obtenido con PT. En este caso la glucosa es consumida más lento que en los datos experimentales, por lo que se generan los efectos ya mencionados en la primera iteración: mayor concentración máxima de etanol, consumo más rápido de sacarosa y producción más rápida de etanol y fructosa. Sin embargo, estas diferencias de velocidades de generación y consumo son menores que en las primeras dos iteraciones. La gran diferencia ocurre en la concentración de levaduras, que sigue descendiendo incluso cuando se termina la sacarosa. Este comportamiento sugiere que aún queda glucosa porque en caso contrario la Ecuación 4.16 se haría cero y la concentración de levaduras permanecería constante.

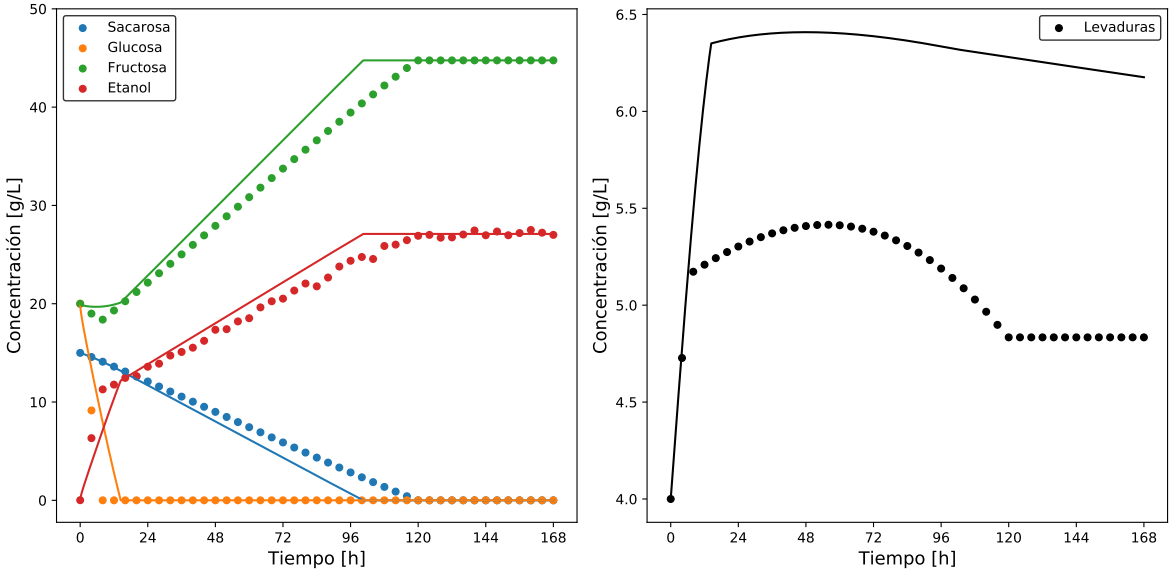


Figura 4.28: Simulación con las C.I. de validación del modelo del mejor individuo obtenido con penalización total.

Un objetivo secundario de la penalización total era disminuir el tiempo de ejecución del algoritmo GP al resolver el sistema de ecuaciones de los individuos que no presentaran la variable deseada. Sin embargo, como se presenta en el gráfico de la Figura 4.29, la mediana del tiempo de ejecución con PT (155 minutos) es mayor que la del caso sin penalización (147 minutos), y muy similar a la del caso de PP (156 minutos), teniendo un rango más amplio de tiempos de ejecución que esta última. En general, al incluir penalización de cualquier tipo la ejecución de GP tiende a tomar más tiempo, lo que se puede explicar, en parte, por la búsqueda que se debe hacer de la variable deseada en cada uno de los individuos. Otra posible causa de este aumento de tiempo es que puede que en los casos de penalización se generen individuos más complejos, que suelen tardar más en ser evaluados [125]. En la Tabla 4.14 se puede apreciar que el mejor individuo en el caso sin penalización es menos complejo que el de PP, que también es menos complejo que el de PT. Por lo general, los resultados de las demás repeticiones de cada caso siguen el mismo patrón, los individuos sin penalización son los que identificaron expresiones menos complejas, seguidos por los de PP y los de PT. Por lo tanto, la complejidad de las expresiones matemáticas puede ser la razón por la que hay repeticiones con penalización total que tardaron más tiempo en ejecutarse en comparación con los otros dos casos a pesar de no resolver el sistema de ecuaciones de los individuos penalizados.

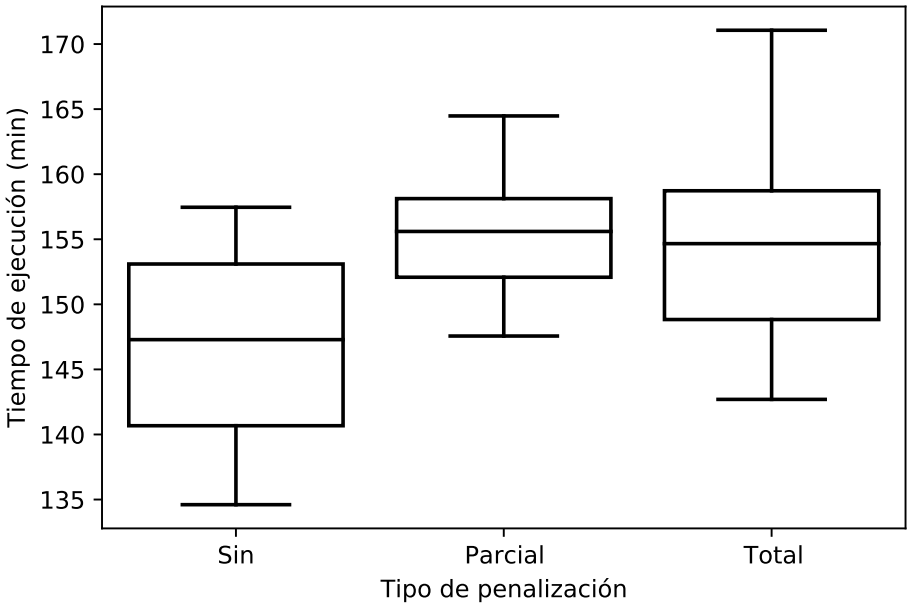


Figura 4.29: Tiempo de ejecución del algoritmo GP en las repeticiones realizadas sin penalización, con penalización parcial y con penalización total.

Otra desventaja de la penalización total es que no permite comparar a los individuos penalizados debido a que se le asigna a todos el mismo fitness. Si estos individuos no pueden ser comparados entre sí es como si en la práctica no existieran, lo que reduce el tamaño efectivo de la población. Aproximadamente un 10 % de los individuos de la última generación de cada repetición con PT corresponden a individuos penalizados. Teniendo en cuenta que a esas alturas del algoritmo la población ya ha convergido, este porcentaje debería ser mayor en las

generaciones más tempranas. Por lo tanto, es posible que existan individuos penalizados que contengan expresiones que son útiles para identificar el bloque faltante, pero que no pasarán esta información a las próximas generaciones por tener un fitness muy pobre.

Lo anterior puede ser el motivo de que se identifiquen modelos más complejos cuando se utiliza la penalización total. El problema de este tipo de modelos es que dificulta el proceso de ajuste de parámetros, porque aumenta el número de constantes que se deben optimizar. Además, si el modelo tiene muchas variables es más probable que ocurra sobreajuste [113]. Como ya se ha mencionado repetidamente en secciones anteriores, para mejorar la generalización del ajuste de parámetros se pueden utilizar técnicas de remuestreo como Jackknife [81], bootstrapping [140] o muestreo aleatorio [112].

4.2.7. Ajuste de parámetros con remuestreo

En esta sección se estudia el efecto del ajuste de parámetros utilizando un método de remuestreo en la capacidad de generalización del modelo propuesto. El método de remuestreo utilizado se basó en la validación cruzada k-fold, en la que se eliminan datos del conjunto de entrenamiento y se ajustan los parámetros con el subconjunto restante. Este proceso se repite k veces y el i-ésimo parámetro final (p_i^*) se calcula a partir del parámetro obtenido utilizando todos los datos (p_i^{SR}) y los obtenidos en los k remuestreos realizados como se muestra en la Ecuación 4.21, donde $p_{i,j}$ corresponde al valor del i-ésimo parámetro obtenido en el remuestreo j [81]. Debido a que se tiene un conjunto de datos extenso se realizaron 10 remuestreos ($k = 10$), eliminando 4 datos en cada remuestreo [147].

$$p_i^* = \frac{1}{k} \cdot \sum_{j=1}^{k} (k \cdot p_i^{SR} - (k-1) \cdot p_{i,j}) \tag{4.21}$$

Se utilizó como ejemplo la expresión matemática del mejor individuo obtenido al ejecutar el algoritmo GP con penalización total porque, de los modelos presentados previamente, este es el más complejo. En la Tabla 4.15 se muestran los resultados del ajuste de parámetros usando la técnica de remuestreo y su comparación con el resultado obtenido sin esta técnica. Se aprecia que al realizar remuestreo se mejora considerablemente el fitness de validación (de 1,87 a 0,147) mientras que el fitness ajustado aumenta (de 0,0938 a 0,727). Pese a la mejora en el fitness de validación este modelo sigue teniendo peor desempeño que el obtenido con penalización parcial, lo que ratifica que ese es el mejor tipo de penalización. De todas maneras se debe tener en cuenta que este análisis se hizo sin eliminar los parámetros no significativos, los resultados obtenidos luego de este proceso se muestran en la Tabla 4.16 (ver Anexo B.4), y de igual forma se aprecia un aumento en el fitness ajustado y un menos fitness de validación al ajustar los parámetros con remuestreo, pero esta diferencia no es tan marcada como la de la Tabla 4.15.

Tabla 4.15: Resultados del ajuste de los parámetros sin remuestreo y con remuestreo.

Tipo de ajuste	**Expresión mejor individuo**	**Fitness ajustado**	**Fitness validación**
Sin remuestreo	$\frac{62,7 \cdot G - 0,0544 \cdot G^2 + 4,03 \cdot E \cdot G^2 + 119 \cdot E \cdot G - 0,229}{7660 \cdot E \cdot G + 0,234}$	0,0938	1,87
K fold	$\frac{1071 \cdot G - 0,304 \cdot G^2 + 0,110 \cdot E \cdot G^2 + 140 \cdot E \cdot G + 0,272}{6623 \cdot E \cdot G + 1377}$	0,727	0,147

Tabla 4.16: Resultados del ajuste de los parámetros sin remuestreo y con remuestreo luego de eliminar parámetros no significativos.

Tipo de ajuste	**Expresión**	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
Sin remuestreo	$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G}{p_6 \cdot E \cdot G + p_7}$	$0,0938$	2,22	$p_1 = 26,3$	0,626	± 107
				$p_3 = 3,87$	0,070	$\pm 4,19$
				$p_4 = 115$	0,090	± 133
				$p_6 = 7357$	0,0739	± 8078
				$p_7 = 0,252$	0,999	± 290
K fold	$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G}{p_6 \cdot E \cdot G + p_7}$	$0,160$	1,43	$p_1 = 129$	0,185	± 192
				$p_3 = 7,76$	0,147	$\pm 10,5$
				$p_4 = 290$	0,144	± 391
				$p_6 = 16727$	0,144	± 22534
				$p_7 = 9,50$	0,999	± 22412

En la Figura 4.30 se muestra la comparación de la simulación de la concentración de etanol a partir de los modelos obtenidos al ajustar los parámetros sin remuestreo y con remuestreo. Este último caso tiene más diferencias con los datos de entrenamiento, pero sin alejarse tanto de ellos, a cambio de ser una representación muy fiel de los datos de validación, diferenciándose de estos principalmente en las primeras 12 horas porque la tasa de producción de etanol es ligeramente mayor. Justamente ese es el objetivo del remuestreo: aumentar la capacidad de generalización del modelo al tener un peor desempeño en la etapa de entrenamiento, pero mejorándolo en la validación. En la Figura 4.31 se muestra que, al igual que en el caso del etanol, las concentraciones de glucosa y fructosa describen casi exactamente los datos experimentales, diferenciándose en las primeras 12 horas porque ambos azúcares se consumen un poco más rápido. La concentración de sacarosa también se asemeja mucho a los datos experimentales y disminuye levemente más rápido a partir de las 72 horas, debido a que hay un poco más de levaduras entre las 24 y 96 horas (cerca de 0,2 (g/L) más en su punto máximo).

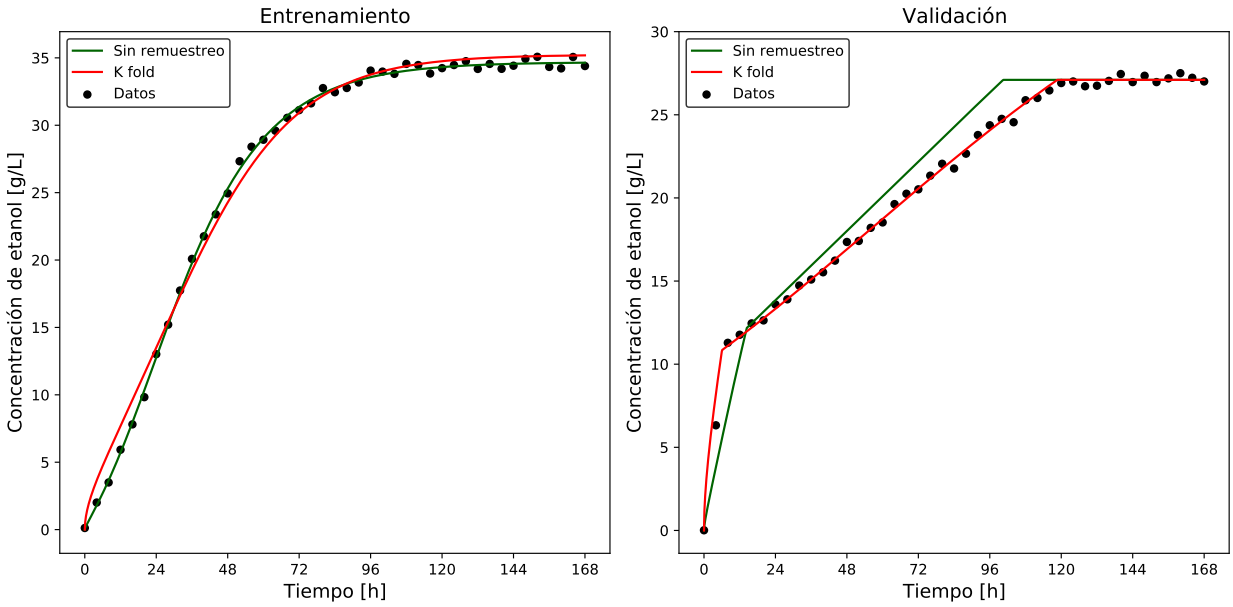


Figura 4.30: Concentración de etanol obtenida al simular los modelos obtenidos al ajustar los parámetros sin y con remuestreo.

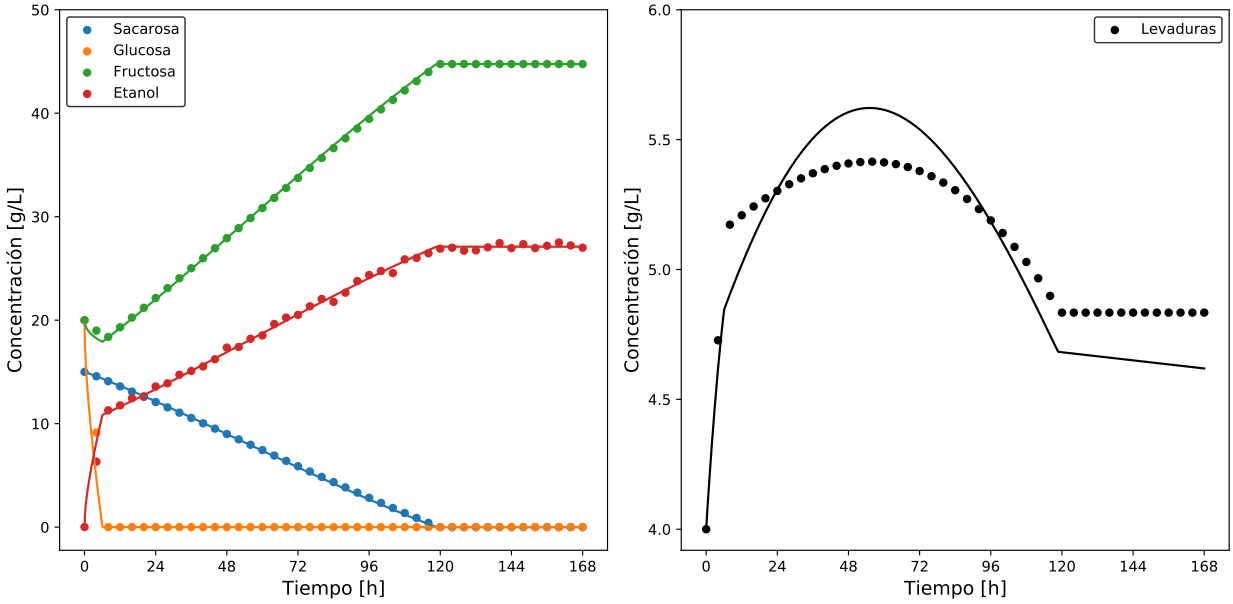


Figura 4.31: Simulación con las C.I. de validación del modelo obtenido luego de ajustar los parámetros con remuestreo.

Un aspecto interesante de los modelos obtenidos con penalización total (sin y con remuestreo) es la evolución de la concentración de levaduras cuando glucosa y sacarosa se acaban. En los gráficos de la Figura 4.32 se muestran las concentraciones de microorganismos obtenidas estos modelos y su comparación con el modelo original. Se observa que la concentración de levaduras en los modelos propuestos sigue disminuyendo después de las 120 horas, qué es el punto a partir del que el modelo original deja de variar. Este descenso es muy lento,

incluso a las 20.000 horas de simulación siguen quedando levaduras. Como ya se discutió, una debilidad del modelo original es que los microorganismos deberían empezar a morir cuando ya no tengan glucosa disponible, por lo que aunque este proceso debería ocurrir más rápido. De todas formas el objetivo de este trabajo es lograr describir los datos experimentales, independiente de los errores del modelo utilizado para generarlos.

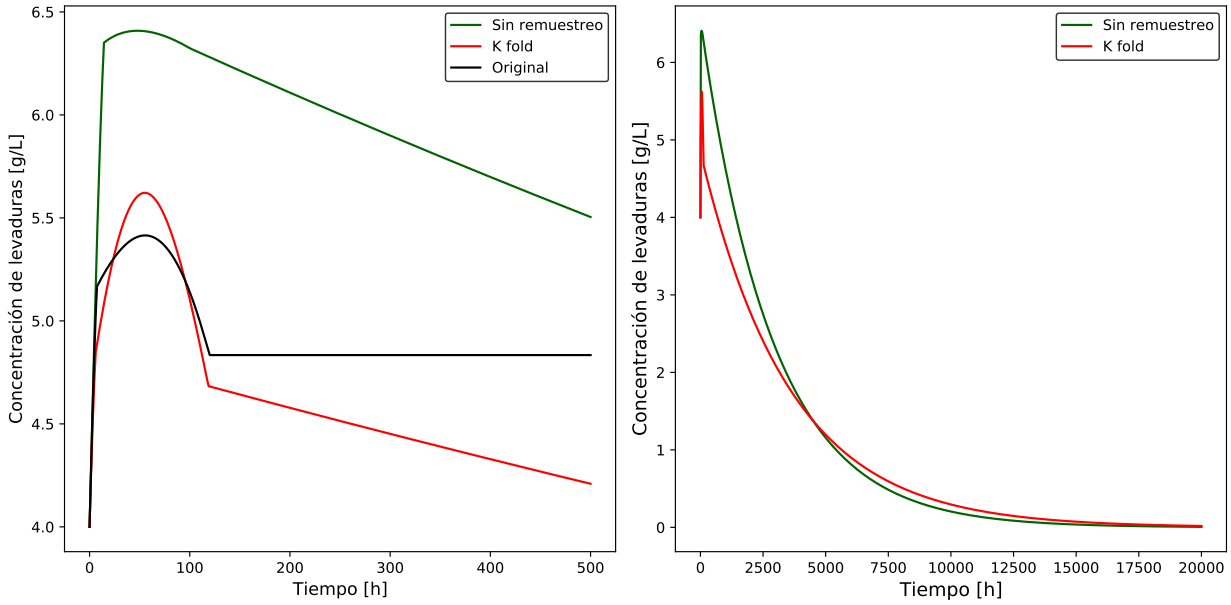


Figura 4.32: Concentración de levaduras obtenidas al simular el modelo obtenido con penalización total (sin y con remuestreo) y su comparación con el modelo original propuesto por Sulieman y col.

En el modelo original cuando la concentración de microorganismos permanece constante porque ya no hay disponibilidad de glucosa, que fue consumida completamente para producir etanol, cuya tasa de formación específica sólo se anula en ausencia de glucosa (Ecuación 4.15). En cambio, la expresión matemática obtenida para q_E luego de ejecutar GP con penalización total (Tabla 4.15) tiene una constante en su numerador: $-0,229$ y $0,272$ para los ajustes de parámetros sin y con remuestreo, respectivamente. Esto hace que todavía quede un poco de glucosa cuando deja de producirse etanol ($q_E = 0$), porque si G fuera 0 q_E no se anularía y la concentración de etanol seguiría variando. Esta concentración de glucosa es la que hace que la concentración de levaduras siga disminuyendo muy lentamente según la Ecuación 4.16 porque la concentración de glucosa es muy cercana a 0.

Entonces, conociendo las ecuaciones y observando los datos experimentales se puede deducir que el numerador de la función obtenida al ejecutar GP con penalización no debe tener la constante en cuestión. Por lo tanto, esta podría eliminarse antes de la etapa de ajuste de parámetros, lo que probablemente mejoraría la capacidad de generalización del modelo. Este es otro ejemplo de cómo el conocimiento previo que se tiene sobre el fenómeno, en este caso las ecuaciones conocidas, permite guiar la búsqueda de la expresión matemática para el bloque faltante.

4.2.8. Análisis del mejor modelo obtenido

Cómo se introdujo en la Sección 2.2.2, este caso de estudio se basa en el sistema de ecuaciones propuesto por Sulieman y col. (2018) [46], representado por las Ecuaciones 4.22-4.27. El objetivo de este caso fue identificar una expresión matemática para la tasa específica de producción de etanol (q_E).

$$\frac{dX}{dt} = \mu \cdot X \tag{4.22}$$

$$\mu = \mu_{max} \cdot (\frac{G}{K_G + G}) \cdot (1 - \frac{E}{E_m}) \tag{4.23}$$

$$\frac{dS}{dt} = -\alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \tag{4.24}$$

$$\frac{dG}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{G/S} - q_E \cdot X \cdot \frac{1}{Y_{X/G}} \tag{4.25}$$

$$\frac{dF}{dt} = \alpha_s \cdot (\frac{S}{K_S + S}) \cdot X \cdot Y_{F/S} - q_E \cdot X \cdot \frac{R}{Y_{X/G}} \tag{4.26}$$

$$\frac{dE}{dt} = q_E \cdot X \cdot \frac{Y_{E/G}}{Y_{X/G}} \cdot (R + 1) \tag{4.27}$$

La mejor expresión para q_E se obtuvo al ejecutar el algortimo GP con penalización parcial por la ausencia de la variable de concentración de etanol (E). Dicha expresión tiene la estructura mostrada en la Ecuación 4.28 y sus parámetros se presentan en la Tabla 4.17.

$$q_E = \frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_3 \cdot G + p_4} \tag{4.28}$$

Tabla 4.17: Parámetros del mejor modelo obtenido.

Parámetro	**p-value**	**IC (95 %)**
$p_1 = 134$	$2{,}53 \cdot 10^{-9}$	$\pm 4,12$
$p_2 = -2,53$	2	$\pm 0,758$
$p_3 = 2651$	$1{,}95 \cdot 10^{-9}$	± 810
$p_4 = 16,4$	0,986	± 1895

El análisis realizado consiste en simular el modelo obtenido con las condiciones iniciales de entrenamiento y con las de validación. Luego, se anula uno de los parámetros (Tabla 4.17) y se simula con ambos conjuntos de condiciones iniciales. Posteriormente, se comparan cada uno de los casos en que se anuló un parámetro con el modelo original. Estas comparaciones se muestran ntre la Figura 4.33 y la Figura 4.40. La Figura 4.33 presenta la simulación con las condiciones iniciales de entrenamiento del caso en que $p_1 = 0$ y la Figura 4.34 presenta el mismo caso con las C.I. de validación.

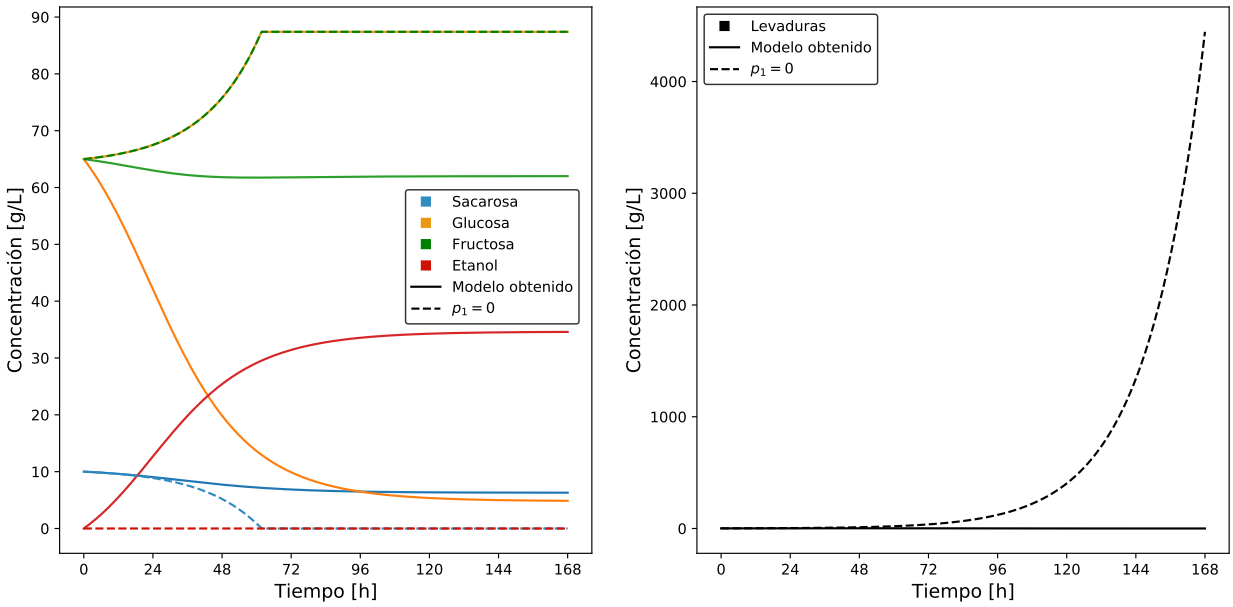


Figura 4.33: Comparación entre el mejor modelo obtenido y dicho modelo con $p_1 = 0$. Simulación con las C.I. de entrenamiento.

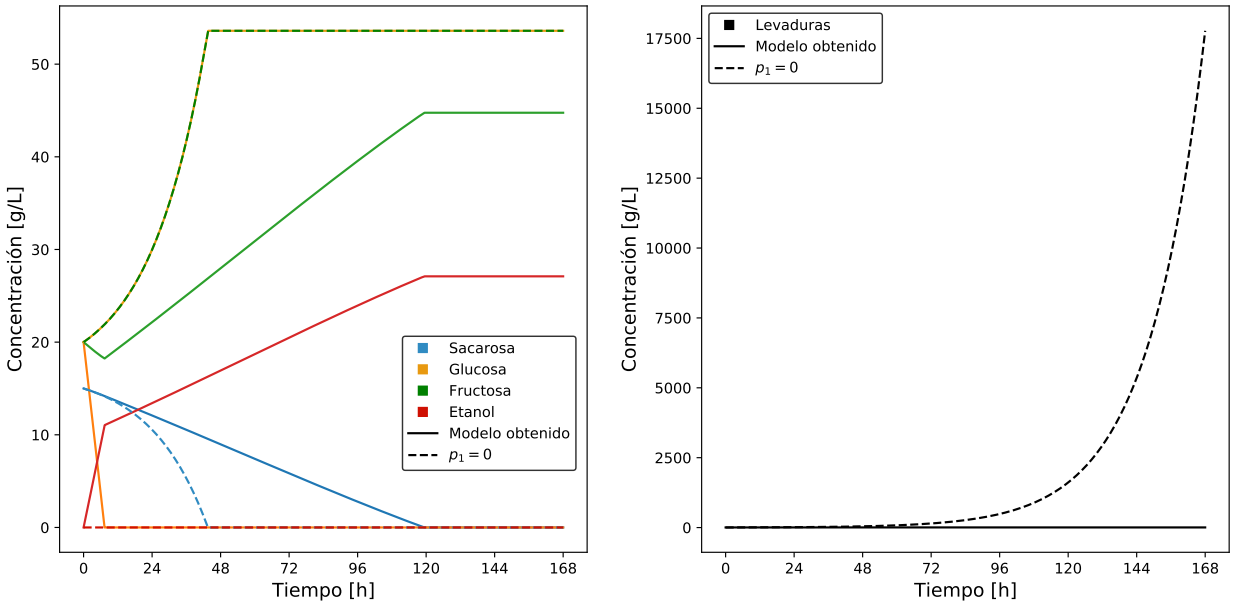


Figura 4.34: Comparación entre el mejor modelo obtenido y dicho modelo con $p_1 = 0$. Simulación con las C.I. de validación.

Tanto en la Figura 4.33 como en la Figura 4.34 se observa que la principal diferencia con el modelo obtenido originalmente es que cuando $p_1 = 0$ no se forma etanol. Como se muestra en la Ecuación 4.29, cuando se anula p_1 la tasa específica de de producción de etanol queda dependiendo directamente de E, que inicialmente es cero en el caso de entrenamiento y en el de validación. Por lo tanto, en ambos casos q_E y E se mantienen nulos por las 168 horas de simulación, y la evolución de las concentraciones de las demás especies es una consecuencia

de ello. Entonces, p_1 es fundamental para la producción de etanol, ya que permite q_E sea mayor que cero cuando la concentración de este producto es nula.

$$q_E = \frac{p_2 \cdot E \cdot G}{p_3 \cdot G + p_4} \tag{4.29}$$

Como se muestra en la Ecuación 4.30, si se anula p_2 la tasa específica de producción de etanol tiene la estructura del modelo de Michaelis-Menten sin inhibición. Entonces, p_2 es el parámetro que da cuenta de la inhibición por producto.

$$q_E = \frac{p_1 \cdot G}{p_3 \cdot G + p_4} \tag{4.30}$$

La Figura 4.35 y la Figura 4.36 presentan las simulaciones del caso en que $p_2 = 0$ con las condiciones iniciales de entrenamiento y con las C.I. de validación, respectivamente. En la Figura 4.35 se observa que la producción de etanol es más rápida, por lo que glucosa y fructosa se consumen más rápido. Otra consecuencia de la mayor producción de E es que se llega antes a la concentración umbral de etanol a partir de la cuál se comienza a producir muerte celular ($E_m = 18\ g/L$, Ecuación 4.23), por lo que la concentración de microorganismos comienza a descender antes, teniendo un comportamiento similar al mostrado en la Figura 4.32 en la Sección 4.2.7, debido a la lenta hidrólisis de sacarosa (recordar que en este modelo la muerte celular depende de la presencia de glucosa, como lo muestra la Ecuación 4.23).

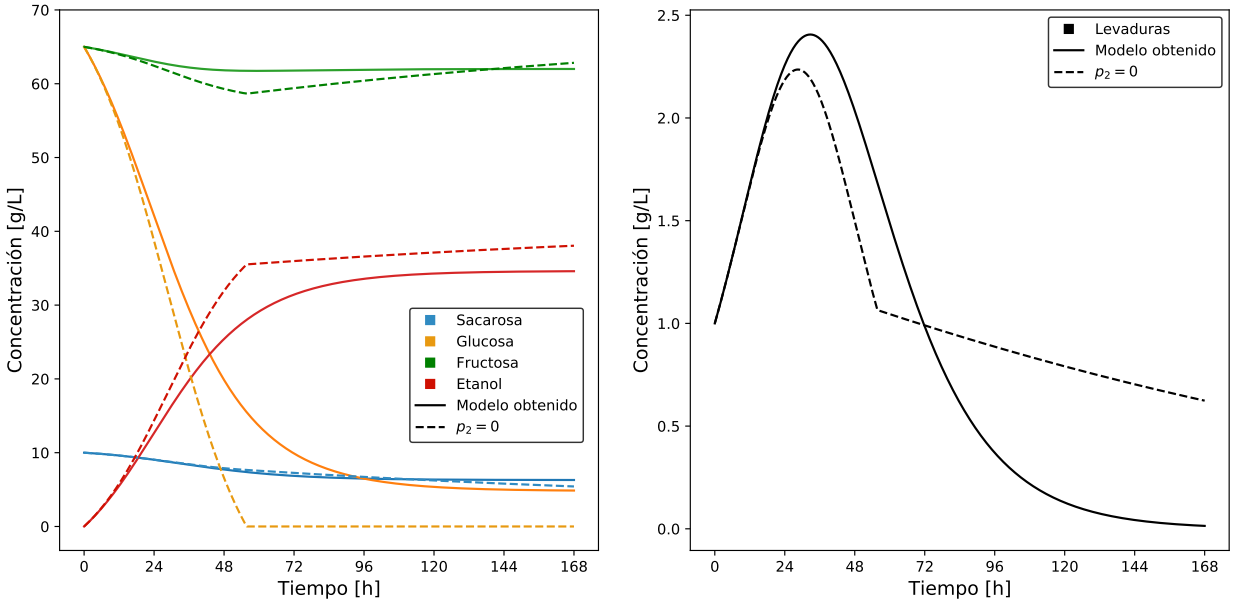


Figura 4.35: Comparación entre el mejor modelo obtenido y dicho modelo con $p_2 = 0$. Simulación con las C.I. de entrenamiento.

En el caso de la Figura 4.36, cuando $p_2 = 0$ la producción inicial de etanol es ligeramente más rápida que en el modelo original. Por lo tanto, el consumo de G y F también es ligeramente mayor, lo que provoca que la glucosa se acabe más rápido, por lo que la concentración de microorganismos es un poco menor en ese instante. Esto genera que la hidrólisis de sa-

carosa en glucosa y fructosa sea ligeramente más lenta luego de ese punto, por lo que E se produce más lento.

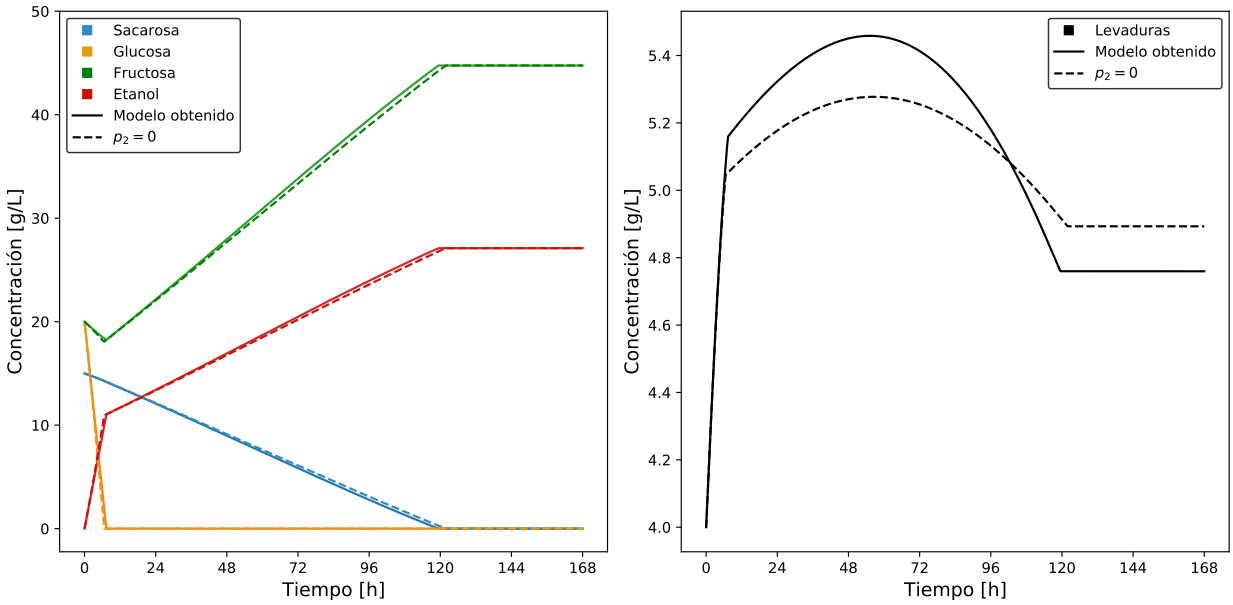


Figura 4.36: Comparación entre el mejor modelo obtenido y dicho modelo con $p_2 = 0$. Simulación con las C.I. de validación.

Como se muestra en la Ecuación 4.31, cuando se anula el parámetro p_3, que originalmente tiene el valor de 2651, la expresión matemática de q_E queda sin variables en su denominador, por lo que es mayor que en el modelo obtenido previamente. Además, como inicialmente la concentración de etanol es cero, al principio de la simulación q_E no es afectado por el término inhibitorio $p_2' \cdot E \cdot G$, por lo que es aún mayor (recordando que p_2' es negativo). El efecto de esto se puede observar en la, que Figura 4.37 presenta la simulación con las condiciones iniciales de entrenamiento del caso en que $p_3 = 0$, y en la Figura 4.38, que presenta dicho caso simulado con las condiciones iniciales de validación.

$$q_E = \frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_4} = \frac{p_1}{p_4} \cdot G + \frac{p_2}{p_4} \cdot E \cdot G = p_1' \cdot G + p_2' \cdot E \cdot G \tag{4.31}$$

Debido a los cambios ocurridos con q_E cuando $p_3 = 0$, tanto para las C.I. de entrenamiento como de validación, la glucosa se consume completamente al inicio de la simulación, lo que provoca un rápido aumento de la concentración de etanol. Luego, a medida que se va produciendo glucosa a partir de la hidrólisis de sacarosa, esta se consume inmediatamente para generar etanol. En la Figura 4.37, E sobrepasa la concentración umbral de etanol a partir de la cuál se comienza a producir muerte celular ($E_m = 18\ g/L$, Ecuación 4.23), por lo que la concentración de microorganismos comienza a descender, y en consecuencia, la hidrólisis de sacarosa se hace más lenta que en el modelo obtenido originalmente.

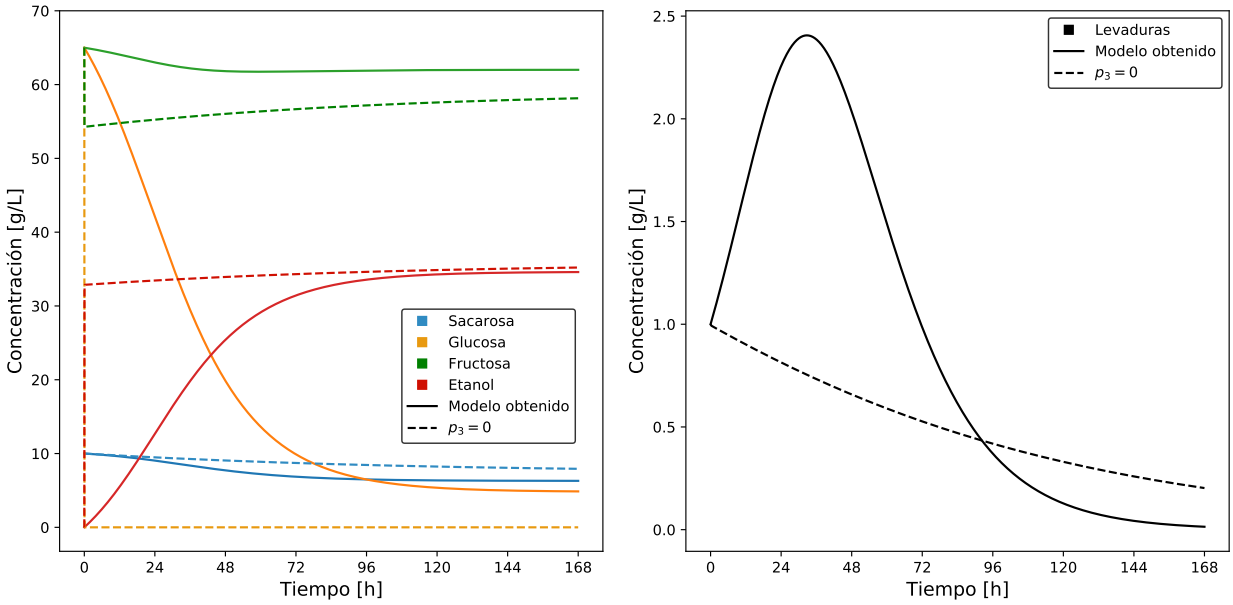


Figura 4.37: Comparación entre el mejor modelo obtenido y dicho modelo con $p_3 = 0$. Simulación con las C.I. de entrenamiento.

En la Figura 4.38, debido a que la glucosa se consume rápidamente para producir etanol la concentración de microorganismos aumenta mucho menos que en el modelo original. Esto hace que la síntesis de glucosa y fructosa a partir de sacarosa sea más lenta, haciendo más lenta la producción de etanol. Cuando se sobrepasan los 18 g/L de etanol, las levaduras comienzan a morir hasta que se acaba la sacarosa y no se puede formar más glucosa.

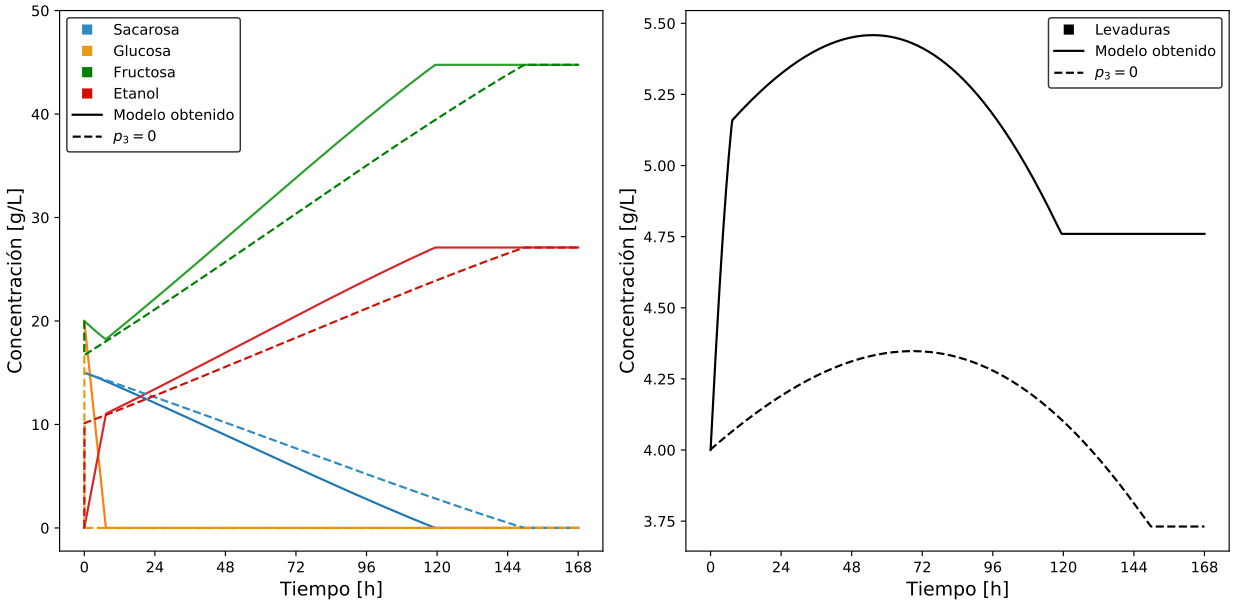


Figura 4.38: Comparación entre el mejor modelo obtenido y dicho modelo con $p_3 = 0$. Simulación con las C.I. de validación.

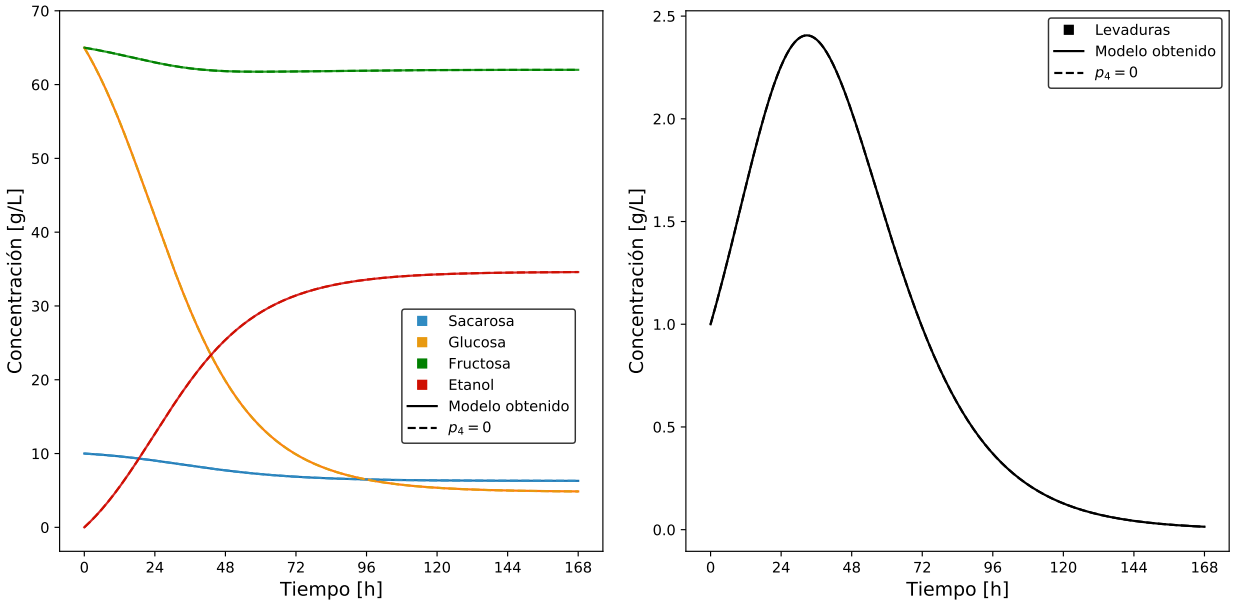


Figura 4.39: Comparación entre el mejor modelo obtenido y dicho modelo con $p_4 = 0$. Simulación con las C.I. de entrenamiento.

La Figura 4.39, la Figura 4.40 y la Figura 4.41 muestran el estudio del efecto de anular el parámetro p_4. En la simulación con condiciones iniciales de entrenamiento (Figura 4.39) se observa que el modelo obtenido originalmente y el modelo con $p_4 = 0$ tienen un comportamiento casi idéntico. Lo mismo ocurre en las primeras 7 horas (aprox.) de la simulación con las C.I. de validación (Figura 4.40), pero como se aprecia de mejor manera en la Figura 4.41, luego de ese tiempo la simulación del caso en que $p_4 = 0$ se detiene, coincidiendo con el momento en que la concentración de glucosa llega a cero. Esto se explica con la Ecuación 4.32, que representa al modelo obtenido con $p_4 = 0$, en el que q_E se indetermina cuando G es cero. Por lo tanto, el principal efecto de p_4 en el modelo obtenido es evitar que q_E se indetermine cuando la concentración de glucosa es cero.

$$q_E = \frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_3 \cdot G} \tag{4.32}$$

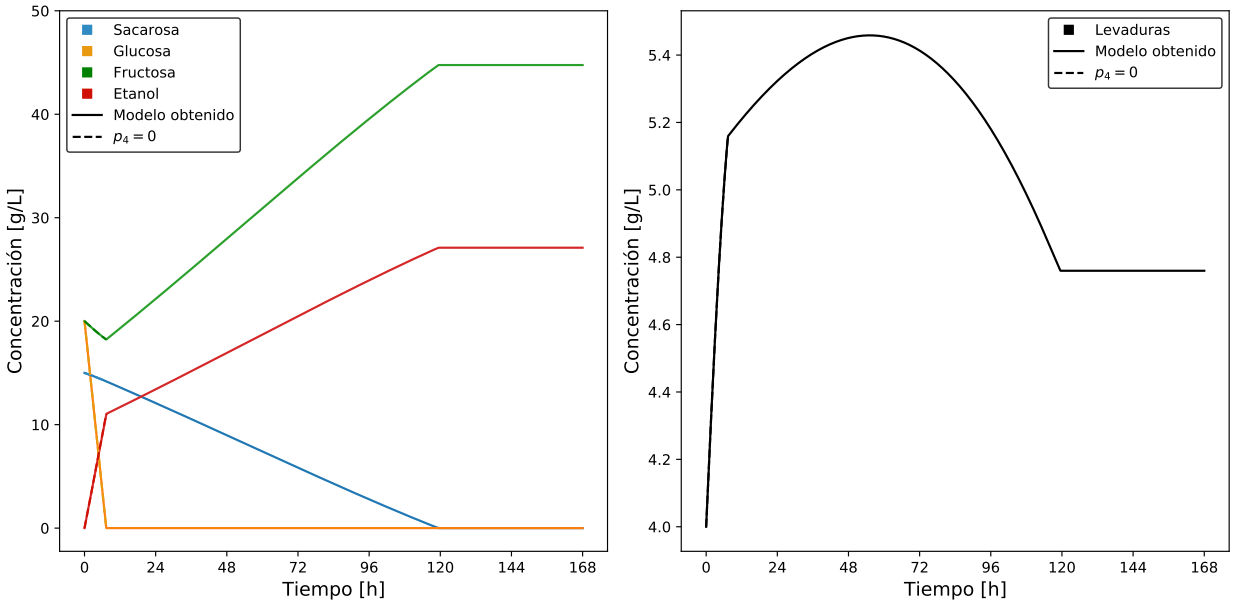


Figura 4.40: Comparación entre el mejor modelo obtenido y dicho modelo con $p_4 = 0$. Simulación con las C.I. de validación.

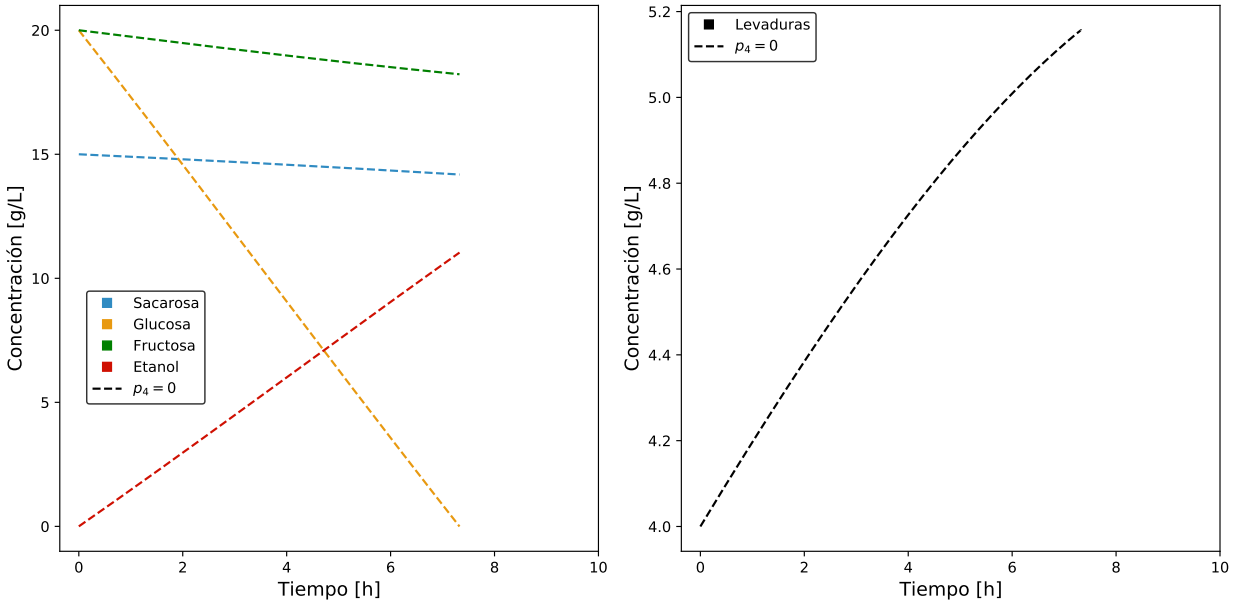


Figura 4.41: Efecto de anular el parámetro p_4 en la simulación del modelo obtenido con la condiciones iniciales de validación.

4.2.9. Limitaciones de la metodología propuesta

Para poder obtener los resultados del caso de estudio 2 fue necesario conocer 5 ecuaciones y 10 constantes para poder obtener resultados. Esta es una clara evidencia de que la principal limitación de la metodología planteada es que sólo se puede identificar un bloque faltante, haciendo que el conocimiento previo sobre el sistema sea un requisito fundamental para el funcionamiento del algoritmo. De igual manera, el sistema estudiado puede ser considerado poco realista, debido a que el bloque desconocido (q_E) tiene exactamente la misma estructura que otra expresión que sí se conoce (μ). Ambas expresiones sólo se diferencian en el valor de un parámetro (E_m y E'_m), por lo que si no se conoce una es muy poco probable que se conozca la otra. Un caso más realista sería considerar tanto q_E y μ como bloques desconocidos, lo que no es realizable con la metodología propuesta. Por lo tanto, los resultados obtenidos demuestran que esta metodología es útil para identificar bloques desconocidos, pero sólo en casos específicos en los que se tiene un amplio conocimiento del modelo, tanto de las ecuaciones como de los valores de los parámetros que lo componen, y sólo se necesita identificar una función específica.

En la metodología propuesta se utilizan dos conjuntos de datos (entrenamiento y validación) obtenidos al hacer dos experimentos con condiciones iniciales distintas. La necesidad de tener este segundo set de datos puede ser una limitante en casos en que por motivos de costos, no disponibilidad de tiempo o dificultades técnicas no sea posible realizar más de un experimento. En caso de no contar con dos conjuntos obtenidos a partir de distintas condiciones se puede intentar identificar el bloque faltante extrayendo un porcentaje de datos de entrenamiento para utilizarlos en la etapa de validación. Sin embargo, en los casos de estudio se evidenció que cuando el modelo obtenido se sobreajusta a los datos de entrenamiento describe de buena manera el fenómeno estudiado en las condiciones en que dichos datos fueron obtenidos, por lo que es posible que no se pueda detectar el sobreajuste con los datos usados para validarlo. Otra solución es recurrir a técnicas de validación cruzada y ajuste de parámetros con remuestreo, que se demostró puede mejorar la capacidad de generalización, pero no se tendría certeza de su efectividad, por no poder medir el error en condiciones distintas a las de entrenamiento. Por lo tanto, la mejor forma de certificar que el modelo obtenido es generalizado es mediante su comparación con un conjunto de validación obtenido operando bajo distintas condiciones.

Otra limitación que también se relaciona con los conjuntos de datos es el error en las condiciones iniciales de validación. Se presentó la optimización de las condiciones iniciales como una manera efectiva de lidiar con el error en las C.I. de entrenamiento, pero esta metodología no puede ser utilizada en la validación de los modelos propuestos, por lo que es muy importante que los datos de validación tengan el menor error que sea posible en sus C.I. para medir la capacidad de generalización de los resultados.

4.2.10. Posibles mejoras a la metodología propuesta

Una posible manera de mejorar esta metodología es utilizar la estrategia de fraccionamiento o partición (*partitioning*), propuesta por Bongard y Lipson (2007), en la etapa de identificación de la estructura del bloque faltante [126]. Cuando se utiliza la partición, en lugar de resolver el sistema de ecuaciones que describe a todas las variables en conjunto, se integra la ecuación que representa a una variable sustituyendo las referencias a las demás variables por los valores experimentales de estas. De esta manera se resuelve la limitación de necesitar toda la información sobre las ecuaciones que describen a las demás variables e incluso es posible identificarlas a todas simultáneamente. Para implementar el fraccionamiento se debe diseñar una nueva función de fitness que integre la ecuación de los individuos reemplazando las demás referencias a otras variables por los datos experimentales correspondientes. La desventaja de este método es que se deben tener datos experimentales para todas las variables del sistema, mientras que en la metodología utilizada basta con tener datos de una variable para calcular el fitness de cada individuo, como en este caso de estudio en que sólo se utilizó la concentración de etanol (igualmente se usó el resto de los datos para estudiar el comportamiento del fenómeno estudiado). Otra desventaja es que se debe editar el funcionamiento de la caja de herramientas GPLAB o implementar el algoritmo desde cero, pues GPLAB está diseñada para identificar sólo una función por repetición.

En caso de necesitar mejorar la capacidad de generalización en la etapa de ejecución del algoritmo GP se modificar la función de evaluación diseñada y agregar alguna técnica de remuestreo, como la técnica de muestreo aleatorio (RST), que consiste en calcular el fitness de los individuos utilizando en cada generación distintos subconjuntos de datos, seleccionados aleatoriamente a partir de los datos de entrenamiento [112]. Otra alternativa es el *backwarding*, que consiste en utilizar dos conjuntos de datos distintos en cada generación, uno para calcular el fitness de los individuos y otro para calcular un fitness de validación del mejor individuo (M.I.) [123]. El fitness de validación se calcula sólo para los individuos que tengan mejor fitness que el M.I., y este último será reemplazado sólo si tiene peor fitness y peor fitness de validación que un nuevo individuo. Esta opción también tiene la limitación descrita en la sección anterior de necesitar más de un conjunto de datos experimentales. Además, requiere que se edite el funcionamiento del algortimo GP en GPLAB, para cambiar su método de selección del mejor individuo. Ambas opciones descritas también pueden ayudar a disminuir la aparición de variables innecesarias, reduciendo la complejidad del modelo.

Capítulo 5

Conclusiones

El objetivo de este trabajo fue desarrollar y evaluar una metodología que permita proponer funciones o bloques faltantes en modelos matemáticos diferenciales de sistemas biológicos utilizando el algoritmo de programación genética (GP). Se diseñó una función de evaluación que para GP que se encarga de formar el sistema de ecuaciones del fenómeno estudiado a partir de la expresión matemática propuesta por cada individuo, resolverlo y comparar los resultados con los datos experimentales.

La validación de los modelos matemáticos propuestos es de vital importancia para asegurarse de su capacidad de generalización y para descartar los casos de sobreajuste. Para esto es necesario contar con conjunto de datos experimentales de validación obtenidos en condiciones distintas que los de entrenamiento.

En el primer caso de estudio, se evidenció que el tamaño de población debe ser los suficientemente amplio como para garantizar la convergencia del algoritmo, pero se debe evitar utilizar poblaciones muy grandes porque pueden hacer que el algoritmo tarde un tiempo innecesariamente largo en ejecutarse. También, se observó que los cambios más significativos al mejor individuo ocurren en las generaciones más tempranas, y que luego de la generación 50 la esto la probabilidad de encontrar mejores soluciones disminuye notablemente. Por lo tanto, es recomendable privilegiar un aumento del tamaño de la población por sobre el número de generaciones en caso de no obtener los resultados deseados en una primera iteración.

En el segundo caso de estudio se tuvo un peor desempeño que en el primero. Fue necesario utilizar la penalización por ausencia de variable para obtener resultados generalizados. La penalización parcial tuvo mejor desempeño que la penalización total. Además, se puede mejorar la capacidad de generalización utilizando remuestreo en el ajuste de parámetros. Otra manera de utilizar conocimiento extra sobre el fenómeno estudiado es la inclusión de una parte conocida del bloque faltante, que permite obtener mejores resultados en menos tiempo.

Se obtuvieron resultados aceptables incluso cuando los datos experimentales presentaron error. En estos casos es necesario conocer las condiciones iniciales sin error de datos de entrenamiento para poder optimizar los parámetros. En caso de no tenerlas pueden incluirse como parámetros a optimizar. Lo que sí es fundamental es que las C.I. de validación no tengan error para poder medir la capacidad de generalización de los modelos propuestos.

La principal limitación de la metodología propuesta es que se necesita conocer las ecuaciones de todas las variables del sistema, sus parámetros y su relación con el bloque faltante para poder aplicarla. Por lo que es utilizable sólo en casos específicos en que se tiene gran nivel de conocimiento sobre el problema, pero se desconoce una función (o bloque faltante).

Finalmente, la aplicación de esta metodología no sólo se restringe a la identificación de modelos diferenciales de sistemas biológicos sino que también puede ser utilizada en problemas de otras áreas del conocimiento o disciplinas.

Bibliografía

[1] FRANCESCHINI, G. y MACCHIETTO, S. 2008. Model-based design of experiments for parameter precision: State of the art. Chemical Engineering Science 63(19):4846-4872.

[2] JUMPER, J., EVANS, R. y PRITZEL, A. 2021. Highly accurate protein structure prediction with AlphaFold. Nature 596:583-589.

[3] HECK, G., PINTRO, V., PEREIRA, R., DE ÁVILA, M., BERNHARDT, N. y DE AZEVEDO, W. 2017. Supervised Machine Learning Methods Applied to Predict Ligand-Binding Affinity. Current Medicinal Chemistry 24(23):2459-2470.

[4] PETSAGKOURAKIS, P., SANDOVAL, I.O., BRADFORD, E., ZHANG, D, y DEL RIO-CHANONA, E.A. 2020. einforcement learning for batch bioprocess optimization. Computers & Chemical Engineering 133:106649.

[5] FARID, S.S., NOVAIS, J., KARRI, S., WASHBROOK, J. y TTITCHENER-HOOKER, N.J. 2000. A Tool for Modeling Strategic Decisions in Cell Culture Manufacturing. Biotechnology Progress 16: 829-836.

[6] PETRIDES, D., KOULOURIS, A. y SILETTI, C. 2003. Throughput analysis and debottlenecking of biomanufacturing facilities. What role can process simulators play? Chimica oggi 20: 22-28.

[7] NICOLETTI, M.C., JAIN, L.C. y GIORDANO, R.C. 2009. Computational Intelligence Techniques as Tools for Bioprocess Modelling, Optimization, Supervision and Control. En: Computational Intelligence Techniques as Tools for Bioprocess Modelling, Optimization, Supervision and Control. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.1-23.

[8] MUSTAFA, M.A., WASHBROOK, J., TITCHENER-HOOKER N.J. y FARID, S.S. 2006. Retrofit Decisions within the Biopharmaceutical Industry. Food and Bioproducts Processing 84(1):84-89.

[9] CHHATRE, S. 2012, Modelling Approaches for Bio-Manufacturing Operations. Advances in Biochemical Engineering/Biotechnology 132:85-107.

[10] LIM, A.C., ZHOU, Y.H., WASHBROOK, J., TITCHENER-HOOKER, N.J., y FARID, S.S. 2004. A decisional-support tool to model the impact of regulatory compliance activities in the biomanufacturing industry. Computers & Chemical Engineering 28(5):727-735.

[11] TITCHENER-HOOKER, N.J., ZHOU, Y.H., HOARE, M. y DUNNILL, P. 2001. Biopharmaceutical process development: Part II, Methods of reducing development

time. Pharmaceutical Technology Europe 13: 68, 70, 72-74.

[12] ZHOU, Y.H., HOLWILL, I.L. y TITCHENER-HOOKER, N.J. 1997. A study of the use of computer simulations for the design of integrated downstream processes. Bioprocess Engineering 16(6):367.

[13] ALLAIRE, G. y CRAIG, A. 2007. Numerical analysis and optimization: An introduction to mathematical modelling and numerical simulation. Estados Unidos, Oxford University Press. 472p.

[14] BENDER, E.A. 1978. An Introduction to Mathematical Modelling. New York, NY, Estados Unidos, John Wiley & Sons Inc. 256p.

[15] DEWASME, L., BOGAERTS, P.M., y VANDE WOUWER, A. 2009. Monitoring of Bioprocesses: Mechanistic and Data-Driven Approaches. En: DO CARMO NICOLETTI, M. y LAKHMI, C.J. (Eds.). Computational Intelligence Techniques for Bioprocess Modelling, Supervision and Control. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.57-97.

[16] STEPHANOPOULOS, G.N., ARISTIDOU, A.A. y NIELSEN, J. 1998. Metabolic engineering : principles and methodologies. Academic Press, 752p.

[17] WEBB, J.L. 1963. Enzyme and Metabolic Inhibitors. New York, NY, Estados Unidos, New York Academic Press. 951p.

[18] ANDREWS, J.F. 1968. A mathematical model for the continuous culture of microorganisms utilizing inhibitory substrates. Biotechnology and Bioengineering 10(6):707-723.

[19] GHOSE, T.K. y TYAGI, R.D. 1979. Rapid ethanol fermentation of cellulose hydrolysate. II. Product and substrate inhibition and optimization of fermentor design. Biotechnology and Bioengineering 21(8):1401-1420.

[20] LEVENSPIEL, O. 1980. The monod equation: A revisit and a generalization to product inhibition situations. Biotechnology and Bioengineering 22(8):1671-1687.

[21] BACAËR, N. 2011. Verhulst and the logistic equation (1838). En: A Short History of Mathematical Population Dynamics. London, Reino Unido. Springer London, pp.35-39.

[22] SHULER, M.L. y KARGI, F. 2001. Bioprocess Engineering: Basic Concepts. 2ª ed. Prentice Hall. 553p.

[23] LEMOS, D.A., SONEGO, J.L.S., BOSCHIERO, M.V.. ARAUJO, E.C.C. Araujo, CRUZ, A.J.G. y BADINO, A.C. 2017. Selection and application of nontoxic solvents in extractive ethanol fermentation. Biochemical Engineering Journal 127: 128-135.

[24] JAIROUN, A.A., AL-HEMYARI, S.S. y SHAHWAN, M. 2020. The pandemic of COVID-19 and its implications for the purity and authenticity of alcohol-based hand sanitizers: The health risks associated with falsified sanitizers and recommendations for regulatory and public health bodies. Research in Social and Administrative Pharmacy 17(1):2050-2051.

[25] RASS-HANSEN, J., FALSIG, H., JØRGENSEN, B. y CHRISTENSEN, C. H. 2007. Bioethanol: fuel or feedstock?. Journal of Chemical Technology & Biotechnology 82(4):329–333.

[26] SUN, Y. y CHENG, J. 2002. Hydrolysis of lignocellulosic materials for ethanol production: a review. Bioresource Technology 83(1):1-11.

[27] ROOZBEHANI, B., MOQADAM, S.I., MIRDRIKVAND, M. y ROSHAN, A.C. 2012. Modeling Direct Ethylene Hydration over Zirconium Tungsten Catalyst: Fundamental of Ethanol Production Using the Biggest Global Ethylene Feeding Pipeline in Iran. Energy and Environment Research 2(2):28-36.

[28] RUIZ, M.I., SANCHEZ, C.I., TORRES, R.G. y MOLINA, D.R. 2011. Enzymatic hydrolysis of cassava starch for production of bioethanol with a colombian wild yeast strain. Journal of the Brazilian Chemical Society 22(12):2337-2343.

[29] DEMİRBAS, A. 2005. Bioethanol from Cellulosic Materials: A Renewable Motor Fuel from Biomass. Energy Sources 27(4):327-337.

[30] TECLU, D., TIVCHEV, G., LAING, M. y WALLIS, M. 2009. Determination of the elemental composition of molasses and its suitability as carbon source for growth of sulphate-reducing bacteria. Journal of Hazardous Materials 161(2-3):1157–1165.

[31] LIN, Y. y TANAKA, S. 2006. Ethanol fermentation from biomass resources: current state and prospects. Applied Microbiology and Biotechnology 69(6):627-642.

[32] MALBAŠA, R., LONČAR, E. y DJURIĆ., M. 2008. Comparison of the Products of Kombucha Fermentation on Sucrose and Molasses. Food Chemistry 106(3):1039–1045.

[33] AVIGAD, G. 1982. Sucrose and Other Disaccharides. En: LOEWUS, F.A. y TANNER, W. Plant Carbohydrates I. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.217–347.

[34] Berthels, n., cordero otero, r., bauer, f., thevelein, j. y pretorius, i. 2004. discrepancy in glucose and fructose utilisation during fermentation by wine yeast strains. FEMS yeast research 4(7):683-689.

[35] PÉREZ, M., LUYTEN, K., MICHEL, R., RIOU, C. y BLONDIN, B. 2005. Analysis of hexose carrier expression during wine fermentation: both low- and high-affinity Hxt transporters are expressed. FEMS Yeast Research 5(4-5):351–361.

[36] TRONCHONI, J., GAMERO, A., ARROYO-LÓPEZ, F.N, BARRIO, E. y QUEROL, A. 2009. Differences in the glucose and fructose consumption profiles in diverse Saccharomyces wine species and their hybrids during grape juice fermentation. International Journal of Food Microbiology 134(3):237–243.

[37] PUTRA, M.D. y ABASAEED, A.E. 2015. Prospective Production of Fructose and Ethanol for Food and Fuel Markets by Selective Fermentation of Date Syrups. Applied Engineering in Agriculture 31(3):497–504.

[38] YU, H., GUO, Y., WU, D., ZHAN, W. y LU, G. 2011. Immobilization of glucose isomerase onto GAMM support for isomerization of glucose to fructose. Journal of Molecular Catalysis B: Enzymatic 72(1-2):73-76.

[39] MATIJAŠEVIĆ, LJ., y VASIĆ-RAČKI, Đ. 2000. Separation of glucose/fructose mixtures: counter-current adsorption system. Biochemical Engineering Journal 4(2):101–106.

[40] ATIYEH, H. y DUVNJAK, Z. 2001. Production of fructose and ethanol from media with high sucrose concentrations by a mutant of Saccharomyces cerevisiae. Journal of

Chemical Technology & Biotechnology 76(10):1017–1022.

[41] COLOWICK, S.P. 1973. The Hexokinases. The Enzymes 9:1-48.

[42] MAITRA, P.K. 1975. Glucokinase from yeast. En: WOOD, W.A. Carbohydrate Metabolism Part C. Elsevier. pp.25–30.

[43] LOBO, Z. y MAITRA, P.K. 1977. Genetics of yeast hexokinase. Genetics 86(4):727-744.

[44] LOBO, Z. y MAITRA, P.K. 1977. Physiological role of glucose-phosphorylating enzymes in Saccharomyces cerevisiae Archives of Biochemistry and Biophysics 182(2):639-645.

[45] CARVALHO, R.S., GOMES, L.H., GONZAGA DO P. FILHO, L. y TAVARES, F.C.A. 2008. Obtaining and selection of hexokinases-less strains of Saccharomyces cerevisiae for production of ethanol and fructose from sucrose. Applied Microbiology and Biotechnology 77(5):1131–1137.

[46] SULIEMAN, A.K., PUTRA, M.D., ABASAEED, A.E., GAILY, M. H., AL-ZAHRANI, S. M. y ZEINELABDEEN, M. A. 2018. Kinetic modeling of the simultaneous production of ethanol and fructose by Saccharomyces cerevisiae. Electronic Journal of Biotechnology 34:1–8.

[47] VAN UDEN, N. 1985. Ethanol Toxicity and Ethanol Tolerance in Yeasts. Annual Reports on Fermentation Processes 8:11–58.

[48] BURKOV, A. 2019. The hundred-page machine learning book. Andriy Burkov.

[49] HURWITZ, J. y KIRSCH, D. 2018. Machine learning for dummies, IBM limited edition. John Wiley & Sons, Inc.

[50] HUANG, T.M., KECMAN, V. y KOPRIVA, I. 2006. Kernel Based Algorithms for Mining Huge Data Sets: Supervised, Semi-supervised, and Unsupervised Learning. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. 260p.

[51] AYODELE, T.O. 2010. Types of learning algorithms. En: ZHANG, Y. (Ed.). New advances in machine learning. IntechOpen. pp.19-46.

[52] HASTIE, T., TIBSHIRANI, R. y FRIEDMAN, J. 2008. Overview of Supervised Learning. En: The Elements of Statistical Learning. New York, NY, Estados Unidos, Springer New York. pp.9–41.

[53] WANG, X., WANG X. y WILKES D.M. 2020. Unsupervised Learning for Data Clustering Based Image Segmentation. En: Machine Learning-based Natural Scene Recognition for Mobile Robot Localization in An Unknown Environment. Singapur, Springer Singapore. pp.63-84.

[54] GRAYBILL, F.A y HARIHARAN, K.I. 1994. Regression Analysis: Concepts and Applications. Duxbury Pr. 701p.

[55] STALPH, P. 2014. Analysis and Design of Machine Learning Techniques: Evolutionary Solutions for Regression, Prediction, and Control Problems. Berlin, Alemania, Springer Vieweg. 155p.

[56] DRAPER, N.R. y SMITH, H. 1998. Applied regression analysis. 3ª ed. New York, NY, Estados Unidos, John Wiley & Sons, Inc. 736p.

[57] POLI, R., LANGDON, W.B. y MCPHEE, N.F. 2008. A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd. 252p.

[58] SCHMIDT, M. y LIPSON, H. 2009.Distilling Free-Form Natural Laws from Experimental Data. Science 324(5923):81-85.

[59] KOZA, J. 1992. Genetic programming : on the programming of computers by means of natural selection. A Bradford Book. 840p.

[60] KARABOGA, D., OZTURK, C., KARABOGA, N. y GORKEMLI, B. 2012. Artificial bee colony programming for symbolic regression. Information Sciences 209: 1-15.

[61] SHIRAKAWA, S., OGINO, S. y NAGAO, T. 2008. Dynamic ant programming for automatic construction of programs. IEEJ Transactions on Electrical and Electronic Engineering 3(5):540-548.

[62] GOLAFSHANI, E.M. 2015. Introduction of Biogeography-Based Programming as a new algorithm for solving problems. Applied Mathematics and Computation 270: 1-12.

[63] VADDIREDDY, H. y SAN, O. 2019. Equation Discovery Using Fast Function Extraction: a Deterministic Symbolic Regression Approach. Fluids 4(2):111.

[64] COMPUTATIONAL INTELLIGENCE: A Methodological Introduction. 2016. Por Rudolf Kruse "et al" (Eds.). London, Reino Unido. Springer-Verlag London. 492p.

[65] REYNOSO-MEZA, G., SANCHIS, J., BLASCO, X. y MART´NEZ, M. 2013. Algoritmos Evolutivos y su empleo en el ajuste de controladores del tipo PID: Estado Actual y Perspectivas. Revista Iberoamericana de Automática e Informática Industrial RIAI 10(3):251-268.

[66] GOLDBERG, D.E. 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. 13ª ed. Addison-Wesley Professional. 432p.

[67] YANG, X.S. 2011. Engineering Optimization: An Introduction with Metaheuristic Applications. Wiley & Sons Inc. 376p.

[68] TOĞAN, V. y DALOĞLU, A.T. 2008. An improved genetic algorithm with initial population strategy and self-adaptive member grouping. Computers & Structures 86(11-12):1204-1218.

[69] ALBA, E. y TOMASSINI, M. 2002. Parallelism and evolutionary algorithms. IEEE Transactions on Evolutionary Computation 6(5):443-462.

[70] GEN, M. y LIN, L. 2008. Genetic Algorithms. En: WAH, W (Ed.). Wiley Encyclopedia of Computer Science and Engineering. Wiley-Interscience. pp.1-15.

[71] LEE, H. y HA, C. 2019. Sustainable Integrated Process Planning and Scheduling Optimization Using a Genetic Algorithm with an Integrated Chromosome Representation. Sustainability 11(2):502.

[72] HOLLAND, J. 1992. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Bradford Books. 232p.

[73] MICHALEWICZ, Z. 1996. Genetic Algorithms + Data Structures = Evolution Programs. 3ª ed. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. 387p.

[74] YANG, X.S. 2014. Nature-Inspired Optimization Algorithms. Elsevier Insights. 263p.

[75] INTRODUCCIÓN A los Algoritmos Genéticos y la Programación Genética. 2010. Por Marcos Gestal "et al" (Eds.). A Coruña, España, Universidade Da Coruña. 76p.

[76] WEIMER, W., NGUYEN, T., LE GOUES, C. y FORREST, S. 2009. Automatically finding patches using genetic programming. En: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (Ed.). 2009 IEEE 31st International Conference on Software Engineering. Piscataway, NJ, Estados Unidos, IEEE. pp.364-374.

[77] KOZA, J. R., BENNETT, F. H., ANDRE, D., KEANE, M. A. y DUNLAP, F. 1997. Automated synthesis of analog electrical circuits by means of genetic programming. IEEE Transactions on Evolutionary Computation 1(2):109–128.

[78] FERNÁNDEZ-BLANCO, E., RIVERO, D., GESTAL, M. y DORADO, J. 2013. Classification of signals by means of Genetic Programming. Soft Computing 17(10):1929–1937.

[79] MEDVET, E., FILLON, C. y BARTOLI, A. 2007. Detection of Web Defacements by means of Genetic Programming. En: THIRD INTERNATIONAL Symposium on Information Assurance and Security. Por Ning Zhang "et al" (Eds.). Piscataway, NJ, Estados Unidos, IEEE pp.227-234.

[80] MCKAY, B., WILLIS, M. y BARTON, G. 1997. Steady-state modelling of chemical process systems using genetic programming. Computers Chemical Engineering 21(9):981–996.

[81] COLINS, A., GERDTZEN, Z.P., NUÑEZ, M.T., y SALGADO, J.C. 2017. Mathematical Modeling of Intestinal Iron Absorption Using Genetic Programming. PLOS ONE 12(1):e0169601.

[82] CHEEMA, J.J.S., SANKPAL, N.V., TAMBE, S.S. y KULKARNI, B.D. 2002. Genetic Programming Assisted Stochastic Optimization Strategies for Optimization of Glucose to Gluconic Acid Fermentation. Biotechnology Progress 18(6):1356–1365.

[83] SUGIMOTO, M., KIKUCHI, S. y TOMITA, M. 2005. Reverse engineering of biochemical equations from time-course data by means of genetic programming. Biosystems 80(2):155–164.

[84] MARENBACH, P. y BETTENHAUSEN, K. D. 1997. Data-driven structured modelling of a biotechnological fed-batch fermentation by means of genetic programming. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 211(5):325–332.

[85] SEDGEWICK, R. y FULLER, J. 2002. Algorithms in Java, Parts 1-4 Fundamentals, data structures, sorting, searching. 3ª ed. Addison-Wesley Professional. 768p.

[86] ROTHLAUF, F. 2006. Representations for Genetic and Evolutionary Algorithms. 2ª ed. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. 325p.

[87] KOZA, J. 1994. Genetic programming as a means for programming computers by natural selection. Statistics and Computing 4(2):87-112.

[88] CHEN. Y. y ABRAHAM, A. 2010. Tree-Structure based Hybrid Computational Intelligence: Theoretical Foundations and Applications. Berlin, Alemania, Springer-Verlag

Berlin Heidelberg. 206p.

[89] LAFOE, R. 2002. Data Structures and Algorithms in Java. 2ª ed. Sams Publishing. 800p.

[90] BAE, S. 2019. JavaScript Data Structures and Algorithms: An Introduction to Understanding and Implementing Core Data Structure and Algorithm Fundamentals. Apress. 378p.

[91] LANGDON, W., POLI, R., MCPHEE, N.R. y KOZA, J. 2008. Genetic Programming: An Introduction and Tutorial, with a Survey of Techniques and Applications. En: FULCHER, J. Computational Intelligence: A Compendium. Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.927-1028.

[92] PEDRAZA, C., OYAGA, J. y GÓMEZ, R. 2013. Síntesis booleana con programación genética paralela en CPU y GPU. Ingenium Revista de la facultad de ingeniería 14: 117.

[93] ALKAFAWEEN, E. 2015. Novel Methods for Enhancing the Performance of Genetic Algorithms. Memoria de Máster en Computer Science. Mu'tah, Jordan, Mu'tah University, Department of Information Technology. 87p.

[94] LAPORTE, G. 1992. The traveling salesman problem: An overview of exact and approximate algorithms. European Journal of Operational Research 59(2):231-247.

[95] YUGAY, O., KIM, I., KIM, B. y KO, F. 2008. Hybrid Genetic Algorithm for Solving Traveling Salesman Problem with Sorted Population. En: THIRD INTERNATIONAL Conference on Convergence and Hybrid Information Technology - ICCIT 2008. Por Paul P. Wang "et al" (Eds.).Piscataway, NJ, Estados Unidos, IEEE. pp.1024-1028.

[96] MAARANEN, H., MIETTINEN, K. y PENTTINEN, A. 2006. On initial populations of a genetic algorithm for continuous optimization problems. Journal of Global Optimization 37(3):405-436.

[97] PULLAN, W. 2003. Adapting the genetic algorithm to the travelling salesman problem. En: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (Ed.). The 2003 Congress on Evolutionary Computation, 2003. CEC '03. Piscataway, NJ, Estados Unidos, IEEE. pp.1029-1035.

[98] AHN, C.W., OH, S. y RAMAKRISHNA, R.S. 2005. On the practical genetic algorithms. En: BEYER, H.G. (Ed.). GECCO '05: Proceedings of the 7th annual conference on Genetic and evolutionary computation. New York, NY, Estados Unidos, Association for Computing Machinery. pp.1583–1584.

[99] MONTANA, D.J. 1995. Strongly Typed Genetic Programming Evolutionary Computation 3(2):199-230.

[100] GARG, A. y TAI, K. 2013. Genetic Programming for Modeling Vibratory Finishing Process: Role of Experimental Designs and Fitness Functions. En: SWARM, EVOLUTIONARY, and Memetic Computing: 4th International Conference, SEMCCO 2013, Chennai, India, December 19-21, 2013, Proceedings, Part II. Por Bijaya Ketan Panigrahi "et al" (Eds.). Basel, Suiza, Springer International Publishing. pp.23-31.

[101] Miller, b.l. y goldberg, d.e. 1995. genetic algorithms, tournament selection, and the

effects of noise. complex systems 9: 193-212.

[102] BLICKLE, T. y THIELE, L. 1995. A Mathematical Analysis of Tournament Selection. En: ESHELMAN, L.J. Proceedings of the 6th International Conference on Genetic Algorithms. San Francisco, CA, Estados Unidos, Morgan Kaufmann Publishers Inc. pp.9-16.

[103] BEASLEY, D., BULL, D.R. y MARTIN, R.R. 1993. An Overview of Genetic Algorithms : Part 1, Fundamentals. University Computing 15(2):56-69.

[104] POPESCU, M.C., POPESCU, L.G y MASTORAKIS, N.E. 2009. Applications of genetic algorithms. WSEAS Transactions on Information Science and Applications 6:1782-1791.

[105] LIPOWSKI, A. y LIPOWSKA, D. 2012. Roulette-Wheel Selection via Stochastic Acceptance. Physica A: Statistical Mechanics and Its Applications 391(6):2193–2196.

[106] THOMAS BÄCK, T., FOGEL, D.B. y MICHALEWICZ, Z. 2000. Evolutionary computation 1: Basic algorithms and operators. Taylor & Francis. 374p.

[107] HAUPT, R.L. 2000. Optimum population size and mutation rate for a simple real genetic algorithm that optimizes array factors. En: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (Ed.). IEEE Antennas and Propagation Society International Symposium 2000: Transmitting Waves of Progress to the Next Millennium. Piscataway, NJ, Estados Unidos, IEEE. pp.1034-1037.

[108] PISZCZ, A. y SOULE, T. 2006. Genetic programming: Optimal population sizes for varying complexity problems. En: CATTOLICO, M. GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation. New York, NY, Estados Unidos, Association for Computing Machinery. pp.953–954.

[109] GENETIC PROGRAMMING: An Introduction. 1997. Por Wolfgang Banzhaf “et al”. San Francisco, CA, Estados Unidos, Morgan Kaufmann Publishers Inc. 496p.

[110] SILVA, S., SILVA, P.J.N y COSTA, E. 2005. Resource-Limited Genetic Programming: Replacing Tree Depth Limits. En: ADAPTIVE AND Natural Computing Algorithms: Proceedings of the International Conference in Coimbra, Portugal, 2005. Por Bernadete Ribeiro “et al” (Eds.). Wien, Austria, Springer-Verlag Wien. pp.243-246.

[111] GONÇALVES, I. y SILVA, S. 2013. Balancing Learning and Overfitting in Genetic Programming with Interleaved Sampling of Training Data. En: GENETIC PROGRAMMING: 16th European Conference, EuroGP 2013, Vienna, Austria, April 3-5, 2013, Proceedings. Por Krzysztof Krawiec “et al” (Eds.). Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.73-84.

[112] GONÇALVES, I., SILVA, S., MELO, J.B. y J.M.B. CARREIRAS. 2012. Random Sampling Technique for Overfitting Control in Genetic Programming. En: GENETIC PROGRAMMING: 15th European Conference, EuroGP 2012, Málaga, Spain, April 11-13, 2012, Proceedings. Por Alberto Moraglio “et al” (Eds.). Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.218-229.

[113] MYUNG, I.J. 2000. The Importance of Complexity in Model Selection. Journal of Mathematical Psychology 44(1):190-204.

[114] AKAIKE, H. 1970. Statistical predictor identification. Annals of the Institute of Statistical Mathematics 22(1):203-217.

[115] AKAIKE, H. 1975. A new look at the statistical model identification. IEEE Transactions on Automatic Control 19(6):716-723.

[116] SCHWARZ, G. 1978. Estimating the Dimension of a Model. The Annals of Statistics 6(2):461-464.

[117] VAPNIK, V.N. 1998. Statistical Learning Theory. Wiley-Interscience, 768p.

[118] CHEN, Q., XUE, B., SHANG, L. y ZHANG, M. 2016. Improving Generalisation of Genetic Programming for Symbolic Regression with Structural Risk Minimisation. En: FRIEDRICH, T. (Ed.). GECCO '16 Companion: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion. New York, NY, Estados Unidos, Association for Computing Machinery. pp.709-716.

[119] CHEN, Q., XUE, B y ZHANG, M. 2020. Rademacher Complexity for Enhancing the Generalization of Genetic Programming for Symbolic Regression. IEEE Transactions on Cybernetics PP(99):1-14.

[120] VANNESCHI, L. y SILVA, S. 2009. Using Operator Equalisation for Prediction of Drug Toxicity with Genetic Programming. En: PROGRESS IN Artificial Intelligence: 14th Portuguese Conference on Artificial Intelligence, EPIA 2009, Aveiro, Portugal, October 12-15, 2009, Proceedings. Por Luís Seabra Lopes "et al" (Eds.). Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.65-76.

[121] CAVARETTA, M.J. y CHELLAPILLA, K. 1999. Data mining using genetic programming: the implications of parsimony on generalization error. En: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (Ed.). CEC099: Proceedings of the 1999 Congress on Evolutionary Computation. Piscataway, NJ, Estados Unidos, IEEE. pp.1330-1337.

[122] ATIF, R.M., MEDERNACH, D. y RYAN, C. 2014. Efficient interleaved sampling of training data in genetic programming. En: IGEL, G. (Ed.). GECCO Comp '14: Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation. New York, NY, Estados Unidos, Association for Computing Machinery. pp.127-128.

[123] ROBILLIARD, D. y FONLUPT, C. 2001. Backwarding: An Overfitting Control for Genetic Programming in a Remote Sensing Application. En: 5th International Conference on Artificial Evolution (Evolution Artificielle) - EA 2001: 29 al 31 de octubre de 2001. Le Creusot, Francia. 373p.

[124] FOREMAN, N. y EVETT, M. 2005. Preventing overfitting in GP with canary functions. En: BEYER, H.G. (Ed.). GECCO '05: Proceedings of the 7th annual conference on Genetic and evolutionary computation New York, NY, Estados Unidos, Association for Computing Machinery. pp.1779-1780.

[125] Leonardo Trujillo, Perla Juárez, Mario García-Valdez, Francisco Vega, and Francisco Chávez. Local search in speciation-based bloat control for genetic programming. *Genetic Programming and Evolvable Machines*, 20:351–384, 2019.

[126] BONGARD, J. y LIPSON, H. 2007. Automated reverse engineering of nonlinear dyna-

mical systems. Proceedings of the National Academy of Sciences 104(24):9943-9948.

[127] DIGNUM, S. y POLI, R. 2008. Operator Equalisation and Bloat Free GP. En: GENETIC PROGRAMMING: 11th European Conference, EuroGP 2008, Naples, Italy, March 26-28, 2008, Proceedings. Por Michael O'Neill "et al" (Eds.). Berlin, Alemania, Springer-Verlag Berlin Heidelberg. pp.110-121.

[128] Numerical Recipes 3rd Edition: The Art of Scientific Computing. 2007. Por William H. Press "et al". 3ª ed. Cambridge, Reino Unido, Cambridge University Press. 1235p.

[129] BYRNE, G.D. y HINDMARSH, A.C. 1987. Stiff ODE solvers: A review of current and coming attractions. Journal of Computational Physics 70(1):1-62.

[130] SOTTAS, G. 1984. Rational Runge-Kutta methods are not suitable for stiff systems of ODEs. Journal of Computational and Applied Mathematics 10(2):169-174.

[131] TRUJILLO, L., Z-FLORES, E., JUÁREZ-SMITH, P., LEGRAND, P., SILVA, S., CASTELLI, M., VANNESCHI, L., SCHÜTZE, O. y MUÑOZ, L. . 2018. Local Search is Underused in Genetic Programming. En: GENETIC PROGRAMMING Theory and Practice XIV. Por Rick Riolo "et al" (Eds.). Basel, Suiza, Springer International Publishing. pp.119-137.

[132] KOMMENDA, M., BURLACU, B., KRONBERGER, G. y AFFENZELLER, M. 2019. Parameter identification for symbolic regression using nonlinear least squares. Genetic Programming and Evolvable Machines 21: 471–501.

[133] Z-FLORES, E., TRUJILLO, L., SCHÜTZE, O. y LEGRAND, P. 2014. Evaluating the Effects of Local Search in Genetic Programming. En: EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V. Por Alexandru-Adrian Tantar "et al" (Eds.) Basel, Suiza, Springer International Publishing. pp.213-228.

[134] ESCOBAR, C.A., WEGNER, D.M, GAUR, A. y MORALES-MENENDEZ, R. 2019. Process-Monitoring-for-Quality—A Model Selection Criterion for Genetic Programming. En: EVOLUTIONARY MULTI-CRITERION optimization: 10th International Conference, EMO 2019, East Lansing, MI, USA, March 10-13, 2019, Proceedings. Por Kalyanmoy Deb "et al" (Eds.). Springer International Publishing. pp.151-164.

[135] SILVA, S. y ALMEIDA, J. 2008. GPLAB - A Genetic Programming Toolbox for MATLAB. En: MATWORKS. MathWorks Nordic MATLAB User Conference 20-21 November 2008, Stockholm. Mathworks. pp.s.p.

[136] LAGARIAS, J. C., REEDS, J. A., WRIGHT, M. H. y WRIGHT, P. E. 1998. Convergence Properties of the Nelder–Mead Simplex Method in Low Dimensions. SIAM Journal on Optimization 9(1):112–147.

[137] DORMAND, J.R. y PRINCE, P.J. 1980. A family of embedded Runge-Kutta formulae. Journal of Computational and Applied Mathematics 6(1):19-26.

[138] POLI, R., MCPHEE, N.F. y VANNESCHI, L. 2008. The impact of population size on code growth in GP. En: KEIJZER, M. GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation. New York, NY, Estados Unidos, Association for Computing Machinery. pp.1275–1282.

[139] GLASNER, E. y WEISS, B. 1993. Sensitive dependence on initial conditions. Nonlinearity 6(6):1067-1075.

[140] DUCHESNE, C. y MACGREGOR, J.F. 2001. Jackknife and bootstrap methods in the identification of dynamic models. Journal of Process Control 11(5):553-564.

[141] ATIYEH, H. y DUVNJAK, Z. 2002. Production of Fructose and Ethanol from Sugar Beet Molasses Using Saccharomyces cerevisiae ATCC 36858. Biotechnology Progress 18(2):234-239., 2002.

[142] PUTRA, M.D., ABASAEED, A.E., ZEINELABDEEN, M.A., GAILY, M.H. y SULIEMAN, A.K. 2014. Selective fermentation of pitted dates by S. cerevisiae for the production of concentrated fructose syrups and ethanol. Journal of Physics: Conference Series 495: 1-4.

[143] VALACH, M., KATRLÍK, J., ŠTURDÍK, E. y GEMEINER, P. 2009. Ethanol Gluconobacter biosensor designed for flow injection analysis. Sensors and Actuators B: Chemical 138(2):581-586.

[144] Wild, r., citterio, d., spichiger, j. y spichiger, u.e. 1996. continuous monitoring of ethanol for bioprocess control by a chemical sensor. journal of biotechnology 50(1):37-46.

[145] VERDUYN, C., ZOMERDIJK, T.P.L., VAN DIJKEN, J.P. y SCHEFFERS, W.A. 1984. Continuous Measurement of Ethanol Production by Aerobic Yeast Suspensions with an Enzyme Electrode. Applied Microbiology and Biotechnology 19(3):181–185.

[146] ZHANG, Q., WU, D., LIN, Y., WANG, X., KONG, H. y TANAKA, S. 2015. Substrate and Product Inhibition on Yeast Performance in Ethanol Fermentation. Energy & Fuels 29(2):1019–1027.

[147] SEN, S., SEZER, E.A., GOKCEOGLU, C. y YAGIZ, S. 2012. On sampling strategies for small and continuous data with the modeling of genetic programming and adaptive neuro-fuzzy inference system. Journal of Intelligent & Fuzzy Systems 23(6):297-304.

Anexo A

Análisis estadístico de los parámetros en resultados del caso de estudio 1

El análisis estadístico de los parámetros en cada uno de los modelos obtenidos en el caso de estudio 1 consistió en obtener su p-value, y luego, en caso de obtener algún valor mayor que 0,5, se realiza nuevamente el ajuste de parámetros, eliminando el parámetro cuyo p-value es mayor a 0,5, si hay más de uno se elimina el mayor. Si el nuevo fitness ajustado es igual o menor que el que se tenía previamente, entonces dicho parámetro puede ser eliminado, en caso contrario se mantiene. El proceso se realiza hasta que no se puedan eliminar parámetros.

A.1. Estudio del efecto del tamaño de la población

Tabla A.1: Análisis estadístico 100 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S + p_2 \cdot X + p_3 \cdot S \cdot X}{p_4 \cdot S + p_5}$	$9,18 \cdot 10^{-8}$	$3,23 \cdot 10^{-6}$	$p_1 = 1,02 \cdot 10^{-5}$	0,716	$\pm 5,64 \cdot 10^{-5}$
			$p_2 = -3,41 \cdot 10^{-8}$	1	$\pm 7,00 \cdot 10^{-5}$
			$p_3 = 0,400$	0	$\pm 9,69 \cdot 10^{-5}$
			$p_4 = 1,00$	0	$\pm 3,10 \cdot 10^{-4}$
			$p_5 = 0,300$	0	$\pm 8,92 \cdot 10^{-4}$
$\frac{p_1 \cdot S + p_3 \cdot S \cdot X}{p_4 \cdot S + p_5}$	$1,99 \cdot 10^{-8}$	$6,89 \cdot 10^{-6}$	$p_1 = 1,04 \cdot 10^{-5}$	0,712	$\pm 5,63 \cdot 10^{-5}$
			$p_3 = 0,400$	0	$\pm 9,66 \cdot 10^{-5}$
			$p_4 = 1,00$	0	$\pm 3,09 \cdot 10^{-4}$
			$p_5 = 0,300$	0	$\pm 8,71 \cdot 10^{-4}$
$\frac{p_3 \cdot S \cdot X}{p_4 \cdot S + p_5}$	$1,74 \cdot 10^{-8}$	$4,61 \cdot 10^{-6}$	$p_3 = 0,400$	0	$\pm 3,29 \cdot 10^{-5}$
			$p_4 = 1,00$	0	$\pm 9,12 \cdot 10^{-5}$
			$p_5 = 0,300$	0	$\pm 2,13 \cdot 10^{-4}$

Tabla A.2: Análisis estadístico 150 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3 \cdot X + p_4}$	$2,09 \cdot 10^{-8}$	$2,97 \cdot 10^{-6}$	$p_1 = 0,400$	0	$\pm 0,0169$
			$p_2 = 1,00$	0	$\pm 0,0503$
			$p_3 = -1,58 \cdot 10^{-7}$	1	$\pm 0,0739$
			$p_4 = 0,300$	0,833	$\pm 2,86$
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_4}$	$1,74 \cdot 10^{-8}$	$4,61 \cdot 10^{-6}$	$p_1 = 0,400$	0	$\pm 3,29 \cdot 10^{-5}$
			$p_2 = 1,00$	0	$\pm 9,12 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,13 \cdot 10^{-4}$

Tabla A.3: Análisis estadístico 500 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot X + p_2 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$1,76 \cdot 10^{-8}$	$5,12 \cdot 10^{-6}$	$p_1 = 2,31 \cdot 10^{-6}$	0,878	$\pm 3,02 \cdot 10^{-5}$
			$p_2 = 0,400$	0	$\pm 3,33 \cdot 10^{-5}$
			$p_3 = 1,00$	0	$\pm 9,25 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,31 \cdot 10^{-4}$
$\frac{p_2 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$1,74 \cdot 10^{-8}$	$4,61 \cdot 10^{-6}$	$p_2 = 0,400$	0	$\pm 3,29 \cdot 10^{-5}$
			$p_3 = 1,00$	0	$\pm 9,12 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,13 \cdot 10^{-4}$

Tabla A.4: Análisis estadístico 750 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot X + p_2 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$1,77 \cdot 10^{-8}$	$5,15 \cdot 10^{-6}$	$p_1 = 6,71 \cdot 10^{-7}$	0,965	$\pm 3,07 \cdot 10^{-5}$
			$p_2 = 0,400$	0	$\pm 3,39 \cdot 10^{-5}$
			$p_3 = 1,00$	0	$\pm 9,40 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,34 \cdot 10^{-4}$
$\frac{p_2 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$1,74 \cdot 10^{-8}$	$4,61 \cdot 10^{-6}$	$p_2 = 0,400$	0	$\pm 3,29 \cdot 10^{-5}$
			$p_3 = 1,00$	0	$\pm 9,12 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,13 \cdot 10^{-4}$

Tabla A.5: Análisis estadístico 1000 individuos.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3 \cdot X + p_4}$	$1,78 \cdot 10^{-8}$	$5,06 \cdot 10^{-6}$	$p_1 = 0,400$	0	$\pm 0,00873$
			$p_2 = 1,00$	0	$\pm 0,0259$
			$p_3 = -4,88 \cdot 10^{-7}$	1	$\pm 0,0381$
			$p_4 = 0,300$	0,682	$\pm 1,48$
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_4}$	$1,74 \cdot 10^{-8}$	$4,61 \cdot 10^{-6}$	$p_1 = 0,400$	0	$\pm 3,29 \cdot 10^{-5}$
			$p_2 = 1,00$	0	$\pm 9,12 \cdot 10^{-5}$
			$p_4 = 0,300$	0	$\pm 2,13 \cdot 10^{-4}$

A.2. Estudio del efecto del error en los datos experimentales

Tabla A.6: Análisis estadístico error $\pm 0,5$.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X + p_4}$	$0,112$	0,0868	$p_1 = 0,736$	$9{,}03 \cdot 10^{-9}$	$\pm 0,202$
			$p_2 = 1,84$	$1{,}16 \cdot 10^{-6}$	$\pm 0,642$
			$p_3 = 0,555$	0,479	$\pm 1,57$
			$p_4 = 0,0253$	0,995	$\pm 8,79$
$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X}$	$0,105$	$0,196$	$p_1 = 0,743$	$7{,}55 \cdot 10^{-15}$	$\pm 0,122$
			$p_2 = 1,87$	$3{,}47 \cdot 10^{-13}$	$\pm 0,349$
			$p_3 = 0,362$	0,467	$\pm 0,997$

Tabla A.7: Análisis estadístico error ± 1.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S + p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,347$	0,393	$p_1 = 0,0584$	0,739	$\pm 0,352$
			$p_2 = 0,798$	0,0151	$\pm 0,634$
			$p_3 = 2,00$	0,0318	$\pm 1,82$
			$p_4 = 0,926$	0,540	$\pm 3,04$
$\frac{p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,347$	$0,357$	$p_2 = 0,316$	0,00310	$\pm 0,202$
			$p_3 = 0,780$	0,00535	$\pm 0,534$
			$p_4 = 0,432$	0,217	$\pm 0,697$

Tabla A.8: Análisis estadístico error $\pm 1,5$.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_3 \cdot S \cdot X + p_4}$	$0,775$	0,827	$p_1 = 1,13$	0,276	$\pm 0,207$
			$p_2 = 2,84$	0,310	$\pm 5,58$
			$p_3 = -6,55 \cdot 10^{-4}$	1,02	$\pm 0,0501$
			$p_4 = 0,847$	0,909	$\pm 15,0$
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S + p_4}$	$0,721$	$0,863$	$p_1 = 0,527$	$9{,}91 \cdot 10^{-7}$	$\pm 0,183$
			$p_2 = 1,31$	$8{,}76 \cdot 10^{-6}$	$\pm 0,517$
			$p_4 = 0,420$	0,621	$\pm 1,71$

Tabla A.9: Análisis estadístico error $\pm 0,5$ con ajuste de condiciones iniciales (X_0 y S_0).

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X + p_4}$	$0,0925$	0,597	$X_0 = 0,896$	$8{,}01 \cdot 10^{-8}$	$\pm 0,269$
			$S_0 = 30,16$	0	$\pm 0,280$
			$p_1 = 0,681$	0	$\pm 0,0530$
			$p_2 = 1,71$	$8{,}21 \cdot 10^{-15}$	$\pm 0,270$
			$p_3 = 0,352$	0,594	$\pm 1,33$
			$p_4 = -4,40$	1,32	$\pm 21,4$
$\frac{p_1 \cdot S \cdot X^2}{p_2 \cdot S \cdot X + p_3 \cdot X}$	$0,0931$	2,14	$X_0 = 0,949$	0	$\pm 0,079$
			$S_0 = 30,12$	0	$\pm 0,187$
			$p_1 = 0,695$	0	$\pm 0,0848$
			$p_2 = 1,72$	$2{,}86 \cdot 10^{-14}$	$\pm 0,288$
			$p_3 = 0,358$	0,570	$\pm 1,27$

Tabla A.10: Análisis estadístico error ± 1 con ajuste de condiciones iniciales (X_0 y S_0).

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S + p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,361$	0,439	$X_0 = 1,13$	0,00143	$\pm 0,662$
			$S_0 = 29,95$	0	$\pm 0,636$
			$p_1 = -0,357$	1,29	$\pm 1,93$
			$p_2 = 1,15$	0,0101	$\pm 0,858$
			$p_3 = 2,81$	0,0261	$\pm 2,45$
			$p_4 = 1,70$	0,460	$\pm 4,62$
$\frac{p_2 \cdot S \cdot X^2}{p_3 \cdot S \cdot X + p_4 \cdot X}$	$0,351$	0,925	$X_0 = 1,01$	$1{,}55 \cdot 10^{-14}$	$\pm 0,165$
			$S_0 = 30,05$	0	$\pm 0,364$
			$p_2 = 1,03$	0,00185	$\pm 0,845$
			$p_3 = 2,54$	0,00345	$\pm 2,35$
			$p_4 = 1,40$	0,432	$\pm 35,8$

Tabla A.11: Análisis estadístico error $\pm 1,5$ con ajuste de condiciones iniciales (X_0 y S_0).

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot S \cdot X}{p_2 \cdot S \cdot X + p_3 \cdot S + p_4}$	$0,640$	9,15	$X_0 = 1,01$	$2{,}42 \cdot 10^{-16}$	$\pm 0,366$
			$S_0 = 30,16$	0	$\pm 0,577$
			$p_1 = 1,46$	0,661	$\pm 6,71$
			$p_2 = 2,79 \cdot 10^{-5}$	0,999	$\pm 0,120$
			$p_3 = 3,49$	0,673	$\pm 16,6$
			$p_4 = 4,82$	0,567	$\pm 16,9$
$\frac{p_1 \cdot S \cdot X}{p_3 \cdot S + p_4}$	$0,640$	9,23	$X_0 = 1,01$	$2{,}99 \cdot 10^{-10}$	$\pm 0,239$
			$S_0 = 30,16$	0	$\pm 0,521$
			$p_1 = 1,48$	0,352	$\pm 3,18$
			$p_3 = 3,53$	0,389	$\pm 8,22$
			$p_4 = 4,89$	0,0894	$\pm 5,69$

Anexo B

Análisis estadístico de los parámetros en resultados del caso de estudio 2

El análisis estadístico de los parámetros en cada uno de los modelos obtenidos en el caso de estudio 1 consistió en obtener su p-value, y luego, en caso de obtener algún valor mayor que 0,5, se realiza nuevamente el ajuste de parámetros, eliminando el parámetro cuyo p-value es mayor a 0,5, si hay más de uno se elimina el mayor. Si el nuevo fitness ajustado es igual o menor que el que se tenía previamente, entonces dicho parámetro puede ser eliminado, en caso contrario se mantiene. El proceso se realiza hasta que no se puedan eliminar parámetros.

B.1. Primera iteración

Tabla B.1: Análisis primera iteración.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G}{p_2 \cdot G + p_3 \cdot G \cdot X + p_4}$	$0,138$	7,19	$p_1 = 2,78$	0,0150	$\pm 2,23$
			$p_2 = 13,5$	0,0381	$\pm 12,8$
			$p_3 = 18,0$	0,0101	$\pm 13,6$
			$p_4 = 985$	0,0133	± 776

B.2. Segunda iteración

Tabla B.2: Análisis segunda iteración.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G + p_2 \cdot G^2}{p_3 \cdot G + p_4 \cdot X + p_5 \cdot G \cdot S}$	$0,0949$	4,05	$p_1 = 2,68$	0,00581	$\pm 2,77$
			$p_2 = 0,0691$	0,396	$\pm 0,161$
			$p_3 = 190$	$6{,}76 \cdot 10^{-4}$	± 108
			$p_4 = 77,0$	0,642	± 327
			$p_5 = -5,36$	1,32	$\pm 25,5$
$\frac{p_1 \cdot G + p_2 \cdot G^2}{p_3 \cdot G + p_4 \cdot X}$	$0,0949$	5,75	$p_1 = 2,17$	0,0421	$\pm 2,09$
			$p_2 = 0,0698$	0,0488	$\pm 0,00694$
			$p_3 = 128$	0,0474	± 126
			$p_4 = 96,0$	0,0474	$\pm 9,48$

B.3. Efecto de la penalización por ausencia de variable

Tabla B.3: Análisis penalización parcial.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G + p_2 \cdot E \cdot G + p_3}{p_4 \cdot G + p_5}$	$0,102$	0,0649	$p_1 = 134$	0,0148	± 107
			$p_2 = -2,53$	1,88	$\pm 3,21$
			$p_3 = -8,47 \cdot 10^{-6}$	1	± 5995
			$p_4 = 2651$	0,0515	± 2668
			$p_5 = 16,4$	1	± 413342
$\frac{p_1 \cdot G + p_3}{p_4 \cdot G + p_5}$	$0,377$	4,78	$p_1 = 167$	0,377	± 374
			$p_3 = -8,47 \cdot 10^{-6}$	1	± 427
			$p_4 = 2869$	0,387	± 6535
			$p_5 = 63582$	0,338	± 130867
$\frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_4 \cdot G + p_5}$	$0,102$	0,0649	$p_1 = 134$	$2,53 \cdot 10^{-9}$	$\pm 4,12$
			$p_2 = -2,53$	2	$\pm 0,758$
			$p_4 = 2651$	$1,95 \cdot 10^{-9}$	± 810
			$p_5 = 16,4$	0,986	± 1895
$\frac{p_1 \cdot G}{p_4 \cdot G + p_5}$	$0,301$	4,78	$p_1 = 1743$	0,321	± 3466
			$p_4 = 29877$	0,320	± 59235
			$p_5 = 662051$	0,315	± 1297670
$\frac{p_1 \cdot G + p_2 \cdot E \cdot G}{p_4 \cdot G}$	$0,301$	4,78	$p_1 = 182$	$1,14 \cdot 10^{-9}$	$\pm 54,7$
			$p_2 = -3,44$	2	$\pm 1,03$
			$p_4 = 3610$	$9,29 \cdot 10^{-10}$	± 1079

En este caso se dejan los parámetros presentes en la tercera fila, puesto que en las siguientes filas se obtiene un fitness ajustado mayor.

Tabla B.4: Análisis penalización total.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G + p_2 \cdot G^2 + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G + p_5}{p_6 \cdot E \cdot G + p_7}$	$0,107$	1,79	$p_1 = 62,7$	0,999	±70656
			$p_2 = -0,0544$	1	±1084
			$p_3 = 4,03$	0,401	±9,46
			$p_4 = 119$	0,888	±1666
			$p_5 = -0,229$	1	±899
			$p_6 = 7660$	0,463	±20593
			$p_7 = 0,234$	0,999	±600
$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G + p_5}{p_6 \cdot E \cdot G + p_7}$	$0,0938$	2,22	$p_1 = 26,3$	0,626	±107
			$p_3 = 3,87$	0,070	±4,19
			$p_4 = 115$	0,090	±133
			$p_5 = -0,334$	1	±5251
			$p_6 = 7357$	0,0739	±8078
			$p_7 = 0,252$	0,999	±290
$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G}{p_6 \cdot E \cdot G + p_7}$	$0,0938$	2,22	$p_1 = 26,3$	0,626	±107
			$p_3 = 3,87$	0,070	±4,19
			$p_4 = 115$	0,090	±133
			$p_6 = 7357$	0,0739	±8078
			$p_7 = 0,252$	0,999	±290

B.4. Ajuste de parámetros con remuestreo

Tabla B.5: Análisis ajuste de parámetros con remuestreo.

Expresión	**Fitness ajustado**	**Fitness validación**	**Parámetro**	**p-value**	**IC (95 %)**
$\frac{p_1 \cdot G + p_2 \cdot G^2 + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G + p_5}{p_6 \cdot E \cdot G + p_7}$	$0,727$	0,147	$p_1 = 1071$	0,843	± 10677
			$p_2 = -0,304$	1	± 174
			$p_3 = 0,110$	0,969	$\pm 5,67$
			$p_4 = 140$	0,437	± 356
			$p_5 = 0,272$	1	± 33441
			$p_6 = 6623$	$2{,}07 \cdot 10^{-4}$	± 3427
			$p_7 = 1377$	0,999	$\pm 1,64 \cdot 10^6$
$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G + p_5}{p_6 \cdot E \cdot G + p_7}$	$0,296$	1,71	$p_1 = 5884$	0,648	± 2544
			$p_3 = 2,39$	0,588	$\pm 8,69$
			$p_4 = 17,1$	0,530	± 537
			$p_5 = -1,06$	1	± 25041
			$p_6 = 8812$	0,525	± 27385
			$p_7 = 13229$	0,986	$\pm 1,46 \cdot 10^6$
$\frac{p_1 \cdot G + p_3 \cdot E \cdot G^2 + p_4 \cdot E \cdot G}{p_6 \cdot E \cdot G + p_7}$	$0,160$	1,43	$p_1 = 129$	0,185	± 192
			$p_3 = 7,76$	0,147	$\pm 10,5$
			$p_4 = 290$	0,144	± 391
			$p_6 = 16727$	0,144	± 22534
			$p_7 = 9,50$	0,999	± 22412