Aplicación de métodos de Monte Carlo en el estudio de redes metabólicas

Pablo Teixeira Manion

Maestría en Investigación de Operaciones

Instituto de Computación, Facultad de Ingeniería

Universidad de la República

Montevideo, Uruguay

Agosto 2019

Tesis de Maestría presentada al Programa de Maestría en Investigación de Operaciones, Facultad de Ingeniería de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Magíster en Investigación de Operaciones.

Director Académico: Dr. Héctor Cancela

Directores de Tesis: Dr. Héctor Cancela, Dr. Luis Acerenza

Tribunal: Dr. Eduardo Fernández, Dr. Andrés Pomi, M.Sc. María E. Urquhart

Glosario

ACHR	Artificial Centering Hit-and-Run
ANII	Agencia Nacional de Investigación e Innovación
C	Vector de concentraciones de metabolitos intracelulares
CM	Centro de Masa
COBRApy	Constraints-Based Reconstruction and Analysis for Python
Δ	Variabilidad de flujo
FBA	Flux Balance Analysis (Análisis de Balance de Flujos)
FVA	Flux Variability Analysis (Análisis de Variabilidad de Flujos)
GLPK	GNU Linear Programming Kit
HR	Hit-and-Run
L	Coeficiente de verosimilitud
LB	Lower bound (límite inferior)
LHS	Latin Hypercube Sampling (Muestreo de Hipercubo Latino)
m	Cantidad de filas de la matriz estequiométrica
MCMC	Markov Chain Monte Carlo
MoMA	Minimization of Metabolic Adjustments (Minimización de Ajustes Metabólicos)
n	Cantidad de columnas de la matriz estequiométrica
PCA	Principal Component Analysis (Análisis de Componentes Principales)
S	Matriz estequiométrica
u	Unidad de flujo, mmol gDW^{-1} h^{-1}
UB	Upper bound (límite superior)
V	Vector de flujos metabólicos
v_{BM}	Flujo de producción de biomasa
v_{GLC}	Flujo de entrada de glucosa
W	Base ortonormal del espacio nulo de la matriz estequiométrica
X	Vector aleatorio

Resumen

Los modelos basados en restricciones permiten estudiar las redes metabólicas de diferentes cepas de microorganismos. Imponiendo la condición de estado estacionario, estos modelos definen un espacio de flujos que puede ser analizado mediante simulaciones de Monte Carlo para, por ejemplo, aproximar su volumen. Este volumen está relacionado con el repertorio de posibles respuestas del microorganismo a cambios en las condiciones externas.

El cálculo del volumen es un problema complejo desde el punto de vista computacional, dado que el espacio de flujos suele desarrollarse en muchas dimensiones, por lo que Monte Carlo estándar no es suficiente para analizar los modelos más grandes. En este trabajo se estudió la posibilidad de usar muestreos por importancia y Hit-and-Run para mejorar los resultados del Monte Carlo estándar. La metodología desarrollada se puso a prueba en un modelo ramificado simple y en un modelo núcleo de E.coli compacto, con 25 reacciones y 17 metabolitos. Se validó este modelo observando que se comporta de manera muy similar al núcleo de E.coli y que además verifica el "growth-flexibility trade-off".

El muestreo por importancia no mejoró los resultados del Monte Carlo estándar, pero su aplicación permitió una mayor comprensión de esta técnica aplicada al muestreo en modelos metabólicos, abriendo líneas de trabajo futuro. Con Hit-and-Run se logró una aproximación al orden de magnitud del volumen del espacio de flujos, que puede resultar útil en modelos de muchas dimensiones.

Se aplicaron estas técnicas para el análisis del núcleo de E.coli compacto, encontrándose resultados relevantes cuyas implicancias biológicas se analizaron. A modo de ejemplo, estudiando los volúmenes con distintas fermentaciones en el modelo, se encontró que no existe una sinergia en ningún par de fermentaciones que permita un mayor aumento en el volumen al estar las dos fermentaciones activadas a la vez. Se simuló también una capacidad máxima en el microorganismo imponiendo una restricción a la suma de los valores absolutos de los flujos, buscando un valor óptimo a partir de un indicador que comprende el volumen y la distancia promedio al centro de masa del espacio de soluciones. Por otro lado, al estudiar la relación entre crecimiento máximo y volumen, se encontró que se requiere reducir la cota inferior del crecimiento microbiano a valores inferiores al 60% del crecimiento máximo para obtener un volumen que no sea despreciable con respecto al volumen máximo (obtenido con cota inferior igual a cero).

Si bien el problema de calcular volúmenes en muchas dimensiones de manera precisa está lejos de ser resuelto, estos y otros resultados encontrados muestran que la aplicación de simulaciones de Monte Carlo permite obtener conclusiones significativas a partir de modelos metabólicos, incluso cuando se trata de modelos simplificados.

Agradecimientos

En primer lugar, agradezco a mis directores de tesis, Dr. Héctor Cancela y Dr. Luis Acerenza, por todo el apoyo que me han dado a lo largo de mis estudios de maestría. Agradezco también al Dr. Eduardo Fernández por sus aportes para la elaboración de la metodología de este trabajo en lo que respecta al uso de la descomposición QR. Finalmente, agradezco a la Agencia Nacional de Investigación e Innovación (ANII) por su apoyo económico mediante el sistema de becas de posgrado.

Contenidos

Índice

Índice de anexos

1. Introducción

En el presente trabajo, se intenta explorar la aplicación de métodos de Monte Carlo en el estudio de redes metabólicas, en conjunto con otras herramientas comúnmente usadas para su análisis. La modelización de las redes metabólicas se hace a partir de los llamados "modelos basados en restricciones", que buscan generar una representación matemática del metabolismo a partir de una matriz estequiométrica y una serie de restricciones en los flujos metabólicos. Como parte de este trabajo, se llevó a cabo la construcción de un modelo metabólico simplificado de E.coli, a partir de la reducción de un modelo más complejo, a los efectos de poder poner a prueba las metodologías desarrolladas.

La parte más importante de esta tesis se enfoca en un problema particular del estudio de los modelos basados en restricciones, que es el cálculo del volumen del espacio de soluciones. Al tratarse de volúmenes en muchas dimensiones, este problema presenta actualmente un desafío desde el punto de vista de la investigación de operaciones, y los avances que puedan hacerse en esta área pueden llevar al desarrollo de herramientas que permitan obtener más información relevante a partir de los modelos basados en restricciones.

El documento está organizado en siete capítulos (de los cuales el primero corresponde a la presente introducción). El segundo capítulo presenta un marco teórico para el trabajo, donde se introducen los modelos basados en restricciones, las técnicas más usadas para su estudio, y un estado del arte donde se presentan algunos trabajos relevantes. El tercer capítulo presenta los objetivos de este trabajo. El cuarto capítulo presenta la metodología, donde se introducen los modelos metabólicos existentes y el software libre que fueron empleados en el trabajo, así como los procedimientos para preparar estos modelos para su uso y las técnicas empleadas para cálculo de volumen. El quinto capítulo presenta los resultados obtenidos, tanto para un modelo ramificado simple como para el modelo metabólico simplificado de E.coli que fue construido. Finalmente, el sexto capítulo presenta las conclusiones y el séptimo presenta la bibliografía.

Los resultados preliminares de este trabajo fueron presentados en la conferencia "13th International Conference in Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing" llevada a cabo en Rennes, Francia del 1 al 6 de julio de 2018 [1]. Asimismo, se espera presentar los resultados en el "II Congreso Nacional de Biociencias" que se llevará a cabo en Montevideo, Uruguay del 4 al 7 de septiembre de 2019.

2. Marco teórico

Este capítulo presenta un marco teórico para el trabajo, donde se intenta brindar un concepto general de los modelos basados en restricciones en conjunto con las técnicas que se usan para obtener información a partir de ellos y los avances que se han hecho en los últimos años. La primera sección introduce el pasaje de una red metabólica a un modelo basado en restricciones, a través de la generación de una matriz estequiométrica, el planteo de un estado estacionario del sistema y la definición de restricciones adicionales. La segunda sección presenta una clasificación de las herramientas que se usan para explorar los modelos basados en restricciones y algunos ejemplos tales como análisis de balance de flujos, análisis de variabilidad de flujos, minimización de ajustes metabólicos y el método de Monte Carlo. Finalmente, la tercera sección presenta un estado del arte donde se seleccionan algunos artículos que corresponden al trabajo de otros autores en la caracterización del espacio de soluciones mediante optimización y mediante el método de Monte Carlo, así como en el cálculo del volumen del espacio de soluciones de un modelo basado en restricciones.

2.1. Modelos basados en restricciones

Los modelos basados en restricciones se centran en la creación de una representación matemática de todas las reacciones metabólicas que ocurren en un organismo. La manera más usual de lograr esta representación es mediante una matriz estequiométrica, la cual almacena todos los coeficientes estequiométricos de cada reacción [2]. Un caso particular del modelado basado en restricciones es aquel en el que solo se consideran las restricciones estequiométricas, llamado modelado estequiométrico. Este tipo de modelado permite evitar las dificultades características del desarrollo de modelos cinéticos, que surgen de la falta de experimentos que permitan obtener medidas de flujos intracelulares. El modelado estequiométrico permite explotar el conocimiento de la estructura del metabolismo celular, sin considerar los procesos cinéticos que aún no son bien conocidos [3].

Cada fila de la matriz estequiométrica representa un único metabolito, y cada columna representa una única reacción. Las entradas de la matriz son los coeficientes estequiométricos de los metabolitos que participan en una determinada reacción. El coeficiente es negativo para un metabolito que actúa como sustrato, y positivo cuando este es un producto de la reacción. Un coeficiente estequiométrico de cero es usado para cada metabolito que no participa en una determinada reacción. La matriz estequiométrica es dispersa, ya que la mayoría de las reacciones metabólicas involucran solo unos pocos metabolitos [4].

El siguiente paso para la generación de estos modelos, una vez que se tiene la matriz estequiométrica definida, consiste en generar un sistema de ecuaciones que actúen como restricciones del modelo. Esto se hace mediante la imposición de un balance de masas en todos los metabolitos de la red metabólica. El balance de masas se define en términos del flujo a través de cada reacción y los coeficientes estequiométricos de la reacción, generando un sistema de ecuaciones diferenciales:

$$\frac{dC}{dt} = S \cdot V$$

donde C es el vector de concentraciones de metabolitos intracelulares, S es la matriz estequiométrica y V es el vector de flujos metabólicos [3]. Los flujos metabólicos se representan generalmente en unidades de milimoles por gramo de peso seco por hora (mmol gDW^{-1} h^{-1}) [5]. En este documento se denominará como "unidad de flujo (u)" al mmol gDW^{-1} h^{-1}, de manera de poder expresar volúmenes de espacios de soluciones en unidades de flujo elevadas a la dimensión del espacio. Asumiendo que la red llega a una condición estable de estado estacionario, las ecuaciones diferenciales se pueden expresar en notación matricial de la siguiente forma:

$$S \cdot V = 0$$

donde S es la matriz estequiométrica y V es el vector de flujos metabólicos. Esta ecuación define un espacio donde se encuentran todas las posibles distribuciones de flujo V. La ecuación restringe las distribuciones que puede alcanzar una red metabólica, pero no predice la distribución de flujos real. Esto es acorde a lo esperable ya que un organismo puede exhibir distintos comportamientos dependiendo, por ejemplo, de las condiciones ambientales. Si S es de rango completo, hay m ecuaciones independientes, una por cada concentración de metabolito. Siendo el número de flujos n típicamente mayor que m, el sistema es indeterminado con $n - m$ grados de libertad [3].

Asimismo, se plantea un límite superior y un límite inferior para cada flujo, que pueden surgir de limitaciones termodinámicas o de otras limitaciones de capacidad de flujo. En particular, el límite inferior se iguala a cero cuando la reacción no es reversible. Esto genera una ecuación para cada flujo de la forma:

$$v_j^{LB} \leq v_j \leq v_j^{UB}$$

donde v_j^{LB} representa el límite inferior de v_j y v_j^{UB} representa el límite superior de v_j.

Las restricciones que se añaden al modelo se pueden clasificar en dos tipos principales: no ajustables y ajustables. Las no ajustables son aquellas que no cambian en el tiempo, tales como aquellas impuestas por la termodinámica (por ejemplo, irreversibilidad de determinados flujos) o capacidades máximas de transporte (que determinan un valor de flujo máximo). Las ajustables son aquellas que dependen de las condiciones ambientales y pueden variar de una célula a otra. Un ejemplo posible de restricciones ajustables son medidas de flujos que se obtienen experimentalmente y luego se imponen al modelo. Cuando se estudian las propiedades invariantes de una red metabólica, solo se hace uso de las restricciones no ajustables, ya que estas siempre se cumplen. Si se usan las restricciones ajustables, las propiedades que se deduzcan son válidas para alguna circunstancia en particular en la que apliquen esas restricciones [3].

La Ilustración 1 resume la construcción de un modelo basado en restricciones a partir de una red metabólica.

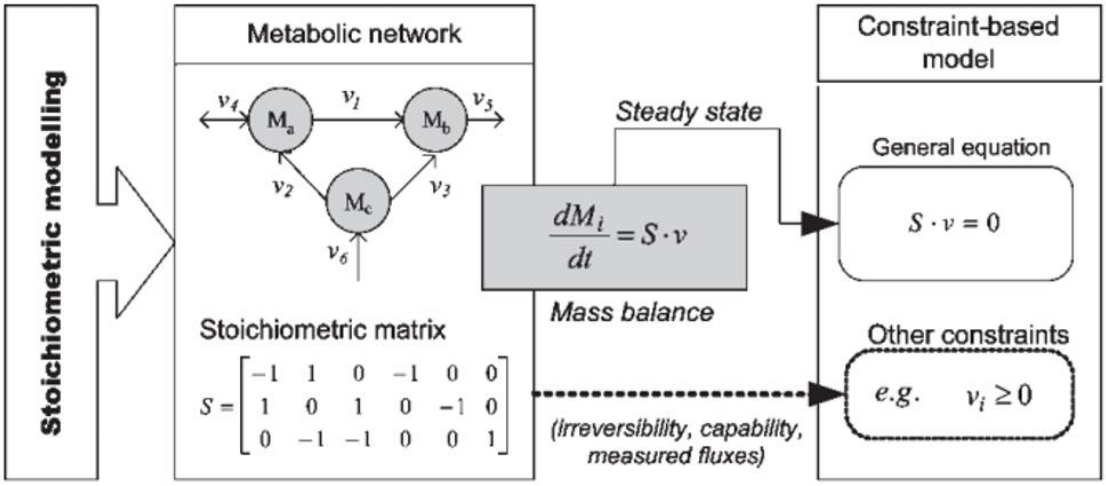


Ilustración 1 – Modelos basados en restricciones, adaptado de [3]

El modelado basado en restricciones se centra en que el organismo modelado está sujeto a restricciones que limitan su comportamiento. En principio, si se conocieran todas las restricciones aplicables a una circunstancia determinada, se podría deducir el estado real de la red metabólica. Sin embargo, esto no es posible actualmente y no hay indicios de que vaya a ser así en un futuro próximo [3]. De todas formas, con una cierta cantidad de restricciones disponibles es posible establecer cuáles distribuciones de flujo pueden o no ser válidas para el organismo modelado. Por ello, se dice que la imposición de restricciones lleva a un espacio de posibles configuraciones de flujo, y no a la predicción de una solución particular.

Dado que hay más reacciones (y por lo tanto flujos) que metabolitos independientes, la solución del estado estacionario para los flujos metabólicos es indeterminada, a menos que se definan restricciones adicionales [6]. Alternativamente, el sistema indeterminado se puede formular como un problema de optimización. La función objetivo, en este caso, dependerá del problema biológico a resolver y del método usado. Si la función objetivo es lineal, el problema se puede resolver como un problema de programación lineal.

Las restricciones estequiométricas restringen el espacio de posibles distribuciones de flujo a un hiperplano, subespacio de $\mathcal{R}^n$. Si los flujos irreversibles se restringen para ser no negativos, el espacio resultante es un cono convexo. Además, si se incluyen suficientes restricciones de capacidad máxima de flujo, el espacio de soluciones corresponde a un politopo [3].

2.2. Técnicas para explorar estos modelos

Según Llaneras y Picó [3], existen distintas metodologías para obtener información a partir de la descripción matemática de un metabolismo que conforma la base del modelado estequiométrico. Cada metodología tiene un propósito propio, utiliza determinadas herramientas matemáticas, y asume una serie de condiciones particulares. Los autores proponen una clasificación de las distintas metodologías en aquellas que se

enfocan en dilucidar propiedades del espacio de soluciones en su totalidad, y aquellas que se enfocan en determinar soluciones particulares dentro de dicho espacio.

Por otro lado, de manera similar a Llaneras y Picó, Lewis et al. [7] clasifican los métodos que estudian todas las posibles soluciones al planteo de las restricciones de una red metabólica como "métodos insesgados" (por ejemplo, análisis del espacio de soluciones mediante Monte Carlo). Estos métodos, a diferencia de los "métodos sesgados" (por ejemplo, FBA), no utilizan una función objetivo para optimizar un determinado flujo, sino que describen las características generales del espacio de soluciones.

El estudio de las características del espacio de soluciones, con métodos insesgados, se puede complementar también con el uso de métodos sesgados, donde se usa una función objetivo para optimizar un determinado flujo. Tal es el caso del trabajo de San Román et al. [8] para E.coli, donde los autores estudian el espacio de soluciones por medio de la definición de una variable a la que llaman "variabilidad de flujo", pero a su vez usan FBA para maximizar la tasa de crecimiento celular, empleando así una combinación de los dos tipos de método.

A continuación se presentan algunos ejemplos de herramientas que se aplican al estudio de modelos basados en restricciones actualmente. Estas herramientas también se discutirán más adelante en la metodología del presente trabajo.

2.2.1. Análisis de Balance de Flujos

El FBA resuelve un problema de programación lineal para encontrar el valor máximo de un flujo (por ejemplo, el flujo que representa la producción de biomasa) bajo ciertas condiciones, según el siguiente planteo formal:

$maximize\ f = v_{BM}$

$s.t.$:

1. $SV = 0$
2. $\alpha_i \leq v_i \leq \beta_i$

2.2.2. Análisis de Variabilidad de Flujos

El FVA consiste en aplicar FBA a todos los flujos del modelo, resolviendo también los problemas de minimización, de manera de obtener un valor máximo y mínimo de funcionamiento para cada reacción en las condiciones determinadas. A continuación se presenta el planteo formal del problema:

$maximize\ (minimize)\ v_i$

$s.t.$:

1. $SV = 0$
2. $\alpha_i \leq v_i \leq \beta_i$
3. $Restricciones\ adicionales\ (e.g. v_{BM} = v_{BM}^{max})$

Se ha demostrado que la variabilidad de flujo depende en gran medida de las condiciones ambientales y la composición de la red metabólica. El desarrollo de formas de cuantificar el repertorio de respuestas metabólicas que puede presentar una red cuando se la somete a distintas condiciones puede brindar conclusiones interesantes acerca del crecimiento y la flexibilidad en la adaptación [9].

2.2.3. Minimización de Ajustes Metabólicos

La MoMA se usa para encontrar la distribución de flujos que minimiza la distancia euclídea a una distribución de flujos de referencia, según el siguiente planteo formal, el cual implica la resolución de un problema de programación cuadrática:

$$minimize\ f = \sqrt{\sum_{i=1}^{n}(v_i - v_i^{ref})^2}$$

$s.t.$:

1. $SV = 0$
2. $\alpha_i \leq v_i \leq \beta_i$
3. $Restricciones\ adicionales\ (e.g. v_{BM} = v_{BM}^{max})$

Esta herramienta puede usarse para modelar matemáticamente un estado sub-óptimo de un microorganismo. Esto puede darse, por ejemplo, al eliminar un determinado gen de un modelo metabólico, donde una hipótesis es que el microorganismo permanece inicialmente lo más cercano posible al óptimo correspondiente a la cepa wild-type (en lugar de ir a un estado óptimo para la nueva situación). Segrè et al. [10] aseguran que MoMA permite aproximar la respuesta a la eliminación de un gen mejor que la búsqueda de un nuevo óptimo mediante FBA. La Ilustración 2 muestra este concepto de forma gráfica.

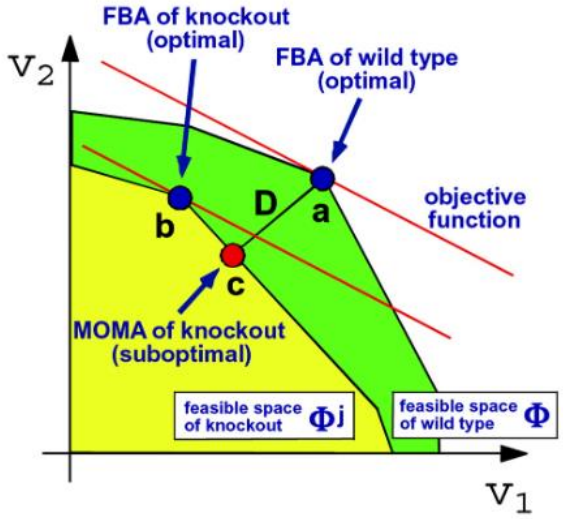


Ilustración 2 – Principio de optimización en MoMA, tomado de [10]

2.2.4. Monte Carlo

Como se mencionó, el hecho de que la solución del estado estacionario para los flujos metabólicos sea indeterminada hace que exista un espacio de soluciones conformado por todos los conjuntos de flujos que cumplen con todas las restricciones del modelo. El muestreo aleatorio de este espacio de soluciones permite estudiar sus características y tamaño [11], por lo que el método de Monte Carlo también se tiene como una de las herramientas que permiten estudiar las características de los modelos metabólicos basados en restricciones.

Una forma de determinar el volumen de un politopo P (por ejemplo, el espacio de soluciones en estado estacionario de un modelo metabólico) es relacionándolo a un volumen de referencia [12]. El volumen de referencia R se elige como una caja que contiene a P. Dentro de R, se toman muestras aleatorias. El método de Monte Carlo directo determina el volumen de P clasificando las muestras en válidas y no válidas según si caen dentro de P o no. Siendo K el número total de muestras y M la cantidad de muestras válidas, un estimador del volumen P se deriva de acuerdo a:

$$vol(P) \approx vol(R) \cdot \frac{M}{K}$$

El uso del método de Monte Carlo y su aplicación al cálculo del volumen del espacio de soluciones de redes metabólicas se ve en más detalle en la sección 4.4.1 de la metodología de este trabajo.

2.3. Estado del arte

En esta sección se presentan trabajos de distintos autores que ilustran el uso de las herramientas que se han empleado en los últimos 15 años para caracterizar el tamaño y la forma del espacio de soluciones de sistemas metabólicos.

2.3.1. Caracterización del espacio de soluciones mediante optimización

En el artículo “Source and regulation of flux variability in Escherichia coli”, San Román et al. [8] definen una variable Δ para el cálculo de la variabilidad de flujo en una red metabólica, permitiendo a los autores estudiar dicha variabilidad de forma cuantitativa en un modelo de E.coli.

$$\Delta = \frac{1}{r} \sum_{i \in R} \frac{v_i^{max} - v_i^{min}}{V_i^{max} - V_i^{min}}$$

Donde r es la cantidad de reacciones internas, v_i^{max} y v_i^{min} son los valores de flujo máximo y flujo mínimo respectivamente para la reacción i bajo las nuevas condiciones, y V_i^{max} y V_i^{min} son los valores de flujo máximo y flujo mínimo respectivamente para la reacción i en las condiciones de referencia.

Se observa que, para condiciones más restringidas que las condiciones de referencia, los valores de Δ van a estar entre 0 y 1. Asimismo, valores mucho menores que 1 implican

que muchas de las reacciones sean afectadas significativamente. Esta variable se usa para cuantificar cambios globales en el flujo inducidos por cambios en las condiciones ambientales.

Adicionalmente, se entiende que la variabilidad de flujo puede provenir de tres fuentes distintas, clasificándosela en: variabilidad interna, variabilidad externa y variabilidad de crecimiento. Por ende, se define un valor de Δ para cada una de ellas. La variabilidad interna es calculada encontrando Δ en un escenario en el que todas las reacciones externas están fijas en el valor correspondiente al valor para crecimiento óptimo. La variabilidad externa se calcula encontrando Δ en un escenario en el que $v_{BM} = v_{BM}^{max}$ y sustrayendo la variabilidad interna. La variabilidad de crecimiento se encuentra relajando esa restricción y sustrayendo la variabilidad externa e interna al valor de Δ encontrado.

En primer lugar, los autores analizan la red mediante FBA. Esto les permite encontrar el crecimiento máximo en la situación de referencia. Posteriormente, aplican FVA para encontrar el valor de flujo máximo y mínimo para cada reacción, bajo las mismas restricciones que usaron para el FBA. Usando la definición presentada, se calcula el valor de Δ, obteniéndose así una variable que actúa como medida de la variabilidad de flujo presente en la red metabólica del modelo. Esta está relacionada al volumen del espacio de soluciones del modelo, ya que un cambio en las condiciones ambientales que reduzca el volumen del espacio de soluciones también disminuye las posibilidades de llevar a cabo una determinada función del metabolismo, haciendo que las respuestas metabólicas sean menos flexibles.

El problema de FVA tiene que ser resuelto, en primer lugar, para las condiciones de referencia, y luego para las demás condiciones que se deseen simular mediante la implementación de restricciones adicionales. Esto permite generar todos los máximos y mínimos de flujo necesarios para el cálculo del valor de Δ. Las restricciones adicionales pueden referirse, por ejemplo, a imponer $v_{BM} = v_{BM}^{max}$ según cuál sea el tipo de variabilidad a calcular (interna, externa, de crecimiento).

Una de las principales conclusiones de este trabajo es que una variabilidad alta requiere que el organismo tenga una tasa de crecimiento menor que el máximo posible para las condiciones externas correspondientes. Una posible interpretación es que los recursos que no se destinan a la producción de biomasa se pueden alocar en otros procesos, incrementando así la variabilidad del metabolismo. Esto puede verse en la siguiente gráfica, tomada de San Román et al., donde se representan los tres componentes de la variabilidad en función de la entrada de glucosa en condiciones aeróbicas. Se observa que la variabilidad interna y la variabilidad externa son despreciables para todo el rango de entrada de glucosa, lo que implica que el sistema no permite variabilidad de flujo. El único caso en el que hay variabilidad es cuando se relaja la restricción de $v_{BM} = v_{BM}^{max}$. En este caso, la variabilidad incrementa con la entrada de glucosa.

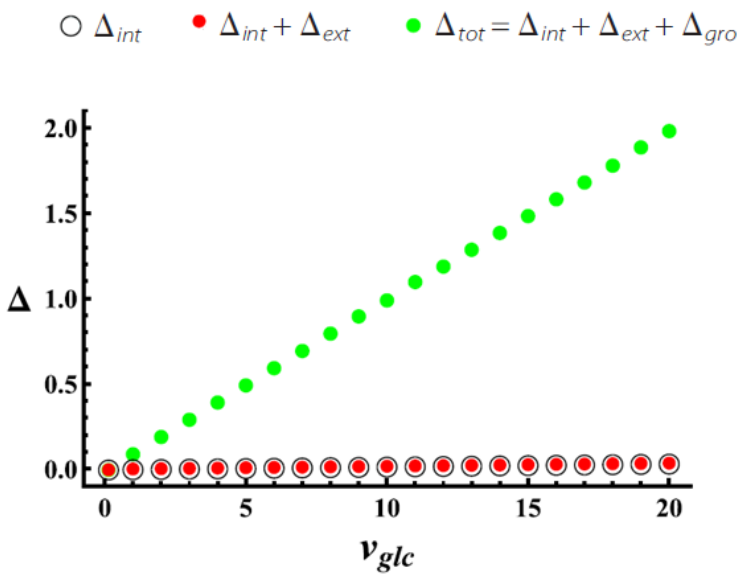


Ilustración 3 – "Components of flux variability vs. glucose uptake in aerobic conditions", tomado de [8]

Aplicando MoMA, los autores observan que, cuando se fija el crecimiento, la cantidad de flujos que cambian es pequeña a lo largo de todo el rango de valores de entrada de glucosa. Por el contrario, cuando se deja el crecimiento libre, la cantidad de flujos que cambian es alta para todo el intervalo. Esto muestra que, bajo el principio de ajuste mínimo de flujos, E.coli presenta una reorganización global del patrón de flujos solo cuando se permite variabilidad en el crecimiento. Los autores concluyen que la reorganización global que ocurre durante la adaptación del organismo a cambios ambientales se obtiene como consecuencia del fenómeno denominado "growth-flexibility trade-off", el cual se explorará también en este trabajo en la sección 5.4.4 mediante el uso del volumen del espacio de soluciones como variable para cuantificar la flexibilidad del metabolismo.

2.3.2. Caracterización del espacio de soluciones mediante Monte Carlo

2.3.2.1. Monte Carlo directo (Price et al. 2004)

El trabajo de Price et al. [11] se sitúa en los inicios del uso de Monte Carlo para el estudio de los modelos basados en restricciones. En este artículo, los autores implementan el método Monte Carlo directo definiendo una manera sencilla de construir una caja de muestreo, la cual será explorada también en el presente trabajo en la sección 4.4.1.2.2.

Los autores usan el método de Monte Carlo para generar muestras del espacio de flujos en estado estacionario y para calcular el volumen de dicho espacio. Para ello, se debe construir un objeto geométrico que contenga al espacio de flujos en estado estacionario, y se sortean puntos aleatorios distribuidos uniformemente dentro de dicho objeto geométrico. El volumen de este objeto debe ser calculable fácilmente. Los autores destacan la importancia de que el objeto se ajuste lo más posible al espacio de flujos, para obtener un porcentaje alto de puntos dentro del objeto que pertenezcan al espacio de flujos. De acuerdo a ellos, un paralelepípedo de igual dimensión que el rango del

espacio nulo de la matriz estequiométrica es un objeto que cumple con estos criterios, y se lo representa por una matriz B donde cada columna de B corresponde a un eje del paralelepípedo.

Los puntos del espacio de soluciones se generan tomando muestras aleatorias dentro del paralelepípedo definido. Las muestras dentro del paralelepípedo se comparan con los límites superior e inferior de los flujos para verificar si pertenecen o no al espacio de soluciones. Si un punto cumple con todas las restricciones, es una solución válida y se lo almacena. Si un punto no satisface todas las restricciones, es excluido. La fracción del total de puntos generados que caen dentro del espacio de soluciones se usa para calcular el volumen de dicho espacio.

Los autores generaron muestras del espacio de flujos en estado estacionario de un modelo *in silico* del metabolismo del glóbulo rojo humano. Los resultados demuestran cómo la metodología presentada por los autores puede usarse para proveer todas las posibles distribuciones para flujos metabólicos en estado estacionario, para guiar la realización de medidas experimentales, y para estudiar los efectos sistémicos de enzimopatías debidas a la baja actividad de alguna enzima.

2.3.2.2. Avances en muestreo (Chaudhary et al. 2016)

Posteriormente, han surgido métodos más avanzados para la aplicación de Monte Carlo a las redes metabólicas que el método de Monte Carlo directo. En este caso, Chaudhary et al. [13] presentan un algoritmo para caracterizar el espacio de flujos en su totalidad en base a la aplicación de FBA. Los autores emplean el muestreo de hipercubo latino (LHS) modificado para muestrear el espacio óptimo, así como el análisis de componentes principales (PCA) para identificar las fuentes de variabilidad en el modelo. Este método fue validado con una red de Saccharomyces cerevisiae y aplicado a una red de Pseudomona aeruginosa. Se demuestra que provee una cobertura más uniforme que un muestreo de Monte Carlo.

En este caso, lo que se quiere no es calcular un volumen, ni encontrar un muestreo que sea representativo estadísticamente, sino que lo que se quiere es caracterizar – en el sentido de cubrir de manera lo más completa posible – el espacio de soluciones y las distintas configuraciones que el mismo abarca.

2.3.3. Cálculo de volumen del espacio de soluciones

2.3.3.1. Monte Carlo directo (Wiback et al. 2004)

La aplicación de estrategias para el cálculo de volúmenes al espacio de soluciones de un modelo metabólico ha sido identificada como un área de interés por diversos motivos, y la investigación en este campo comienza en el año 2004. Uno de los primeros esfuerzos documentados en esta temática, entonces, corresponde al trabajo de Wiback et al. [14]. En dicho trabajo, los autores indican que, hasta la fecha, el estudio de modelos basados en restricciones de redes metabólicas se había centrado en usar programación lineal para optimizar las distribuciones de flujo bajo distintas condiciones. Por tanto, la idea de definir el tamaño y la forma del espacio de soluciones en estado estacionario se presenta aquí como una innovación. En particular, los autores usan muestreos aleatorios por Monte Carlo para caracterizar el tamaño del espacio de soluciones de la red metabólica del glóbulo rojo humano.

Uno de los motivos por los que es importante tener una medida del tamaño del espacio de soluciones en estado estacionario es para ver en qué proporción dicho espacio es reducido al añadir nuevas restricciones a un modelo, que pueden provenir de nuevos datos respecto a la regulación genética o mediciones de flujos metabólicos.

Dada la dificultad de calcular un hipervolumen de muchas dimensiones de manera exacta y las limitaciones computacionales asociadas a esto, los autores sugieren el uso de muestreos de Monte Carlo como una forma de cuantificar el espacio de soluciones en estado estacionario con respecto a un espacio de referencia. En este sentido, se toman puntos aleatorios dentro del espacio de referencia y se calcula la fracción de ellos que cumple con todas las restricciones. La conclusión de los autores es que el método de Monte Carlo funciona de manera excelente a lo hora de calcular dicha fracción cuando se estudian sistemas de bajas dimensiones.

Es necesario tener en cuenta la importancia de determinar un espacio de referencia que permita hacer el muestreo de forma efectiva. Inicialmente, los autores determinaron el politopo de referencia a partir de las cotas máximas y mínimas para los flujos. Sin embargo, esto se volvió problemático al incrementar las dimensiones, ya que el espacio de referencia se volvió demasiado grande en comparación con el espacio de soluciones como para obtener puntos factibles. Por lo tanto, los autores emplearon otros métodos para definir el politopo de referencia que les permitiera acercarse más al espacio de soluciones. De todas formas, para sistemas de muchas dimensiones, esto no hizo la diferencia y no les fue posible encontrar puntos factibles. De acuerdo a los autores, el método de Monte Carlo empleado de esta forma no puede usarse para calcular el hipervolumen absoluto del espacio de soluciones. El único uso que se menciona (para sistemas de muchas dimensiones) es el de calcular cambios relativos en un volumen de referencia al cambiar algunas cotas de flujos máximos y mínimos.

El modelo que usan los autores es el del glóbulo rojo humano. Este modelo contiene 39 metabolitos, 32 reacciones metabólicas internas, y 19 reacciones de intercambio. Para este estudio, todas las reacciones se tomaron como irreversibles, de forma de reducir la dimensión del problema para que el método de Monte Carlo pudiera encontrar puntos válidos en un tiempo razonable. Incluso con esta modificación, los autores indican que llevó más de una semana en un equipo Dell Dimension 8200 para generar 250000 puntos válidos. Sin la modificación, no llegaron a encontrar puntos válidos.

De acuerdo a los autores, este trabajo representa el primer intento de cuantificar el espacio y la forma del espacio de soluciones en estado estacionario. Para menos de 10 dimensiones, les fue posible calcular el volumen exacto del espacio de manera analítica. Para más dimensiones, el cálculo analítico del volumen exacto no se pudo realizar debido a limitaciones computacionales.

El método Monte Carlo fue validado en un sistema de bajas dimensiones para el cual el volumen exacto del politopo pudo ser calculado analíticamente. Se graficaron las distribuciones de valores aleatorios de flujo para cada reacción, lo que permitió analizar la forma del politopo. Las distribuciones de flujo permitieron predecir la reducción en tamaño del espacio de soluciones reduciendo el upper bound de una reacción: el método de Monte Carlo se aplicó a la red metabólica del eritrocito para determinar el tamaño relativo de su espacio de soluciones en estado estacionario. Se mostró que el upper

bound de cada reacción tiene un efecto distinto en el tamaño del espacio de soluciones. La reducción del mismo varió entre un 7% y un 96% dependiendo de cuál fuera la reacción cuyo upper bound se redujo en un 10%. También se mostró que el método de Monte Carlo no llega a cubrir satisfactoriamente las regiones más angostas del espacio de soluciones, no logrando hacer un muestreo sobre todo el rango de los valores de cada flujo.

El método de Monte Carlo directo es ineficiente para calcular el volumen de un politopo en altas dimensiones. Actualmente, se está trabajando en el desarrollo de métodos que permiten lograr cajas de muestreo más ajustadas para obtener una mayor cantidad de muestras válidas. De acuerdo a los autores, el método de Monte Carlo directo solo es aplicable a problemas de hasta un máximo de 15 dimensiones.

2.3.3.2. Avances en uso de muestreos (Latzko et al. 2012)

El espacio de soluciones en estado estacionario de una red metabólica representa un politopo convexo. Determinar el tamaño y la forma de este espacio puede dar información acerca de los procesos metabólicos que ocurren dentro de una célula. En problemas realistas, el politopo suele ser de muchas dimensiones. Dadas las dificultades computacionales que implica el cálculo exacto de su volumen, los métodos de Monte Carlo se pueden usar para obtener una aproximación al mismo. En problemas de más de 20 dimensiones, no alcanza con una aplicación básica de Monte Carlo, sino que se deben emplear metodologías más sofisticadas. En este trabajo, Latzko et al. [12] proponen un método de Markov Chain Monte Carlo, usando el algoritmo Hit-and-Run.

La mayor limitación de una aplicación básica del método de Monte Carlo es que el volumen del politopo se vuelve muy pequeño con respecto al volumen de referencia rápidamente al incrementar las dimensiones del problema. Esto resulta en bajas probabilidades de generar puntos que caigan dentro del politopo.

La estimación del volumen del espacio de soluciones tiene el potencial para convertirse en una nueva herramienta para el análisis de modelos basados en restricciones. El volumen se puede interpretar como una caracterización de la flexibilidad del comportamiento celular.

El algoritmo Hit-and-Run permite evadir el problema de generar solo puntos fuera del politopo, y permite tomar muestras dentro del mismo que tienden asintóticamente a una distribución uniforme. El algoritmo parte de un punto inicial dentro del politopo, genera una dirección aleatoria a partir de una distribución uniforme, y se mueve en esa dirección en ambos sentidos hasta que alguna de las restricciones sea violada. Posteriormente, toma un punto entre esos dos extremos para construir una nueva iteración. Cuantas más muestras se tomen, más se acercan a una distribución uniforme. Este algoritmo es descrito con mayor detalle en el presente trabajo en la sección 4.4.3.

Los autores también presentan el método de Fok-Crevier para la estimación de volúmenes, donde se toman muestras sobre la superficie del politopo (en lugar de dentro de él).

Los autores demuestran que el estimador de Markov Chain Monte Carlo converge en tiempo polinomial, mientras que el tiempo computacional para el método simple de Monte Carlo y el de Fok-Crevier crece de manera exponencial con las dimensiones del

problema. Los métodos de Fok-Crevier y Markov Chain Monte Carlo convergen asintóticamente al volumen real del politopo. Los autores encuentran que, para un politopo de 70 dimensiones, lleva 5.9 horas en dos Intel Xeons con 12 cores para acercarse al volumen con un error del 1%. En comparación con los métodos de Wiback et al. [14], que están limitados por la dimensionalidad del problema, los métodos presentados en este trabajo pueden usarse para problemas realistas en un tiempo razonable.

3. Objetivos

El objetivo principal de este trabajo consiste en explorar el uso de distintas herramientas para el estudio de los modelos metabólicos basados en restricciones, en particular aquellas herramientas que se basan en métodos de Monte Carlo. Asimismo, se busca desarrollar un modelo que represente de forma compacta el núcleo de E.coli, que tenga bajo número de dimensiones y cuyo comportamiento sea similar al del modelo núcleo de E.coli, con la finalidad de poner a prueba las distintas herramientas presentadas.

En lo que respecta a las herramientas de análisis, se detallan los siguientes objetivos particulares:

1. Analizar la factibilidad de aplicar Monte Carlo directo al estudio del volumen del espacio de soluciones de los modelos metabólicos basados en restricciones para distintos tamaños de modelo
2. Introducir el uso de muestreos por importancia al cálculo de dichos volúmenes, analizando cuáles distribuciones son más convenientes y cómo emplearlas
3. Desarrollar una forma de aplicar el algoritmo “Hit-and-Run” al cálculo de dichos volúmenes

Respecto al desarrollo y análisis del modelo núcleo compacto de E.coli, los objetivos particulares son los siguientes:

1. Construir el modelo núcleo compacto de E.coli y validarlo mediante la comparación de su comportamiento con el del modelo núcleo, aplicando distintas herramientas de análisis
2. Aplicar los distintos métodos de cálculo de volumen del espacio de soluciones a este modelo
3. Analizar el comportamiento del modelo frente a distintas situaciones, como por ejemplo el cierre de las distintas fermentaciones, la fijación de las fermentaciones en determinados valores, o la imposición de una restricción de suma de valores absolutos de flujos máxima
4. Estudiar la existencia del fenómeno de “growth-volume trade-off” y compararlo con el fenómeno de “growth-flexibility trade-off” descrito en el marco teórico

4. Metodología

Este capítulo presenta las distintas herramientas utilizadas en el desarrollo de este trabajo, incluyendo los modelos que se usaron como base para el análisis y los algoritmos empleados. En la primera sección, se presentan los modelos metabólicos a partir de los cuales se trabajó. Se intentó aplicar, en primer lugar, todas las metodologías a un modelo simple, para luego adaptarlas a sistemas de más dimensiones. En la segunda sección, se presenta el software libre utilizado en el trabajo. En la tercera sección, se introducen algunos procedimientos para preparar estos modelos para su uso. En la cuarta sección, se introduce el muestreo de los espacios de soluciones de redes metabólicas en estado estacionario aplicando el método de Monte Carlo de manera directa, y se discute junto con las dificultades que esto conlleva al aumentar la dimensionalidad de los espacios. También se introducen otros métodos más avanzados de muestreo, como el muestreo por importancia y el algoritmo "Hit-and-Run".

4.1. Modelos metabólicos

A continuación se presentan los modelos metabólicos que se estudiaron en el curso de este trabajo.

4.1.1. Modelo ramificado simple

4.1.1.1. Introducción

El primer modelo estudiado se trata de una ramificación simple, donde hay un flujo de entrada (V_3) y dos flujos de salida (V_1 y V_2), involucrando a un solo metabolito (A). Este fue el escenario más sencillo que se pudo pensar para lograr entender los procedimientos a implementar. Este mismo modelo es también estudiado por Price et al. [11] para dar un ejemplo de la metodología que los autores usan. La Ilustración 4 presenta este modelo, y la Tabla 1 presenta la matriz estequiométrica correspondiente.

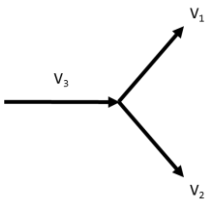


Ilustración 4 – Modelo ramificado simple

Tabla 1 – Matriz estequiométrica del modelo ramificado simple

	V_1	V_2	V_3
A	-1	-1	1

Las cotas inferiores y superiores para los tres flujos del modelo se presentan en la Tabla 2.

Tabla 2 – Cotas inferiores (LB) y superiores (UB) del modelo ramificado simple

	LB	UB
V_1	0	6
V_2	0	8
V_3	0	10

El modelo planteado resultó de gran utilidad a la hora de elaborar las distintas metodologías y elucidar las dificultades que se presentaron, ya que su espacio de soluciones en estado estacionario es de dos dimensiones, y por lo tanto es muy fácil de calcular y visualizar gráficamente.

4.1.1.2. Estudio analítico

El primer ejercicio que se llevó a cabo para comprender el alcance del problema a tratar en este trabajo fue el de calcular el volumen del espacio de soluciones del modelo ramificado simple. Como se mencionó, este modelo es de gran utilidad para una buena comprensión de las metodologías, dado que la baja dimensionalidad del sistema permite una representación gráfica de su espacio de soluciones.

Las siguientes ilustraciones presentan la construcción del espacio de soluciones de este modelo, según las restricciones presentadas en el apartado anterior. Dado que el único flujo de entrada al sistema, V_3, tiene cota superior 10 u, y que no existen flujos reversibles, se elige hacer la representación dentro de un cubo de lado 10 u, donde cada dimensión representa uno de los flujos.

En la Ilustración 5 se presenta el resultado de aplicar la matriz estequiométrica para restringir el espacio de soluciones, sin aplicar todavía las restricciones adicionales de capacidad de flujo (LB y UB). Esta matriz indica un balance para el único metabolito del sistema, llamado metabolito A, que implica que la suma de V_1 y V_2 debe ser igual a V_3. Esto restringe el espacio de soluciones al plano presentado en verde, ubicado dentro del cubo de lado 10 u.

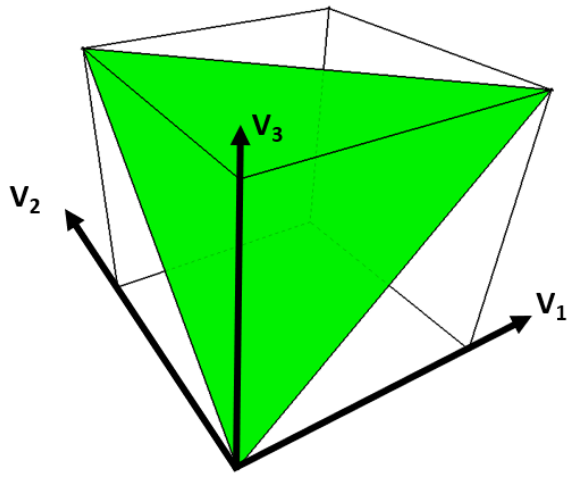


Ilustración 5 – Plano del espacio de soluciones del modelo

Posteriormente, se aplican las restricciones de capacidad de flujo al espacio de soluciones, que cortan el plano presentado. Estas restricciones hacen que V_1 no supere el

valor de 6, V_2 no supere el valor de 8 y V_3 no supere el valor de 10 (lo cual ya estaba representado a la hora de elegir el cubo de lado 10 para la visualización de este espacio). El resultado se presenta en la Ilustración 6, donde las partes en gris quedan ahora fuera del espacio de soluciones.

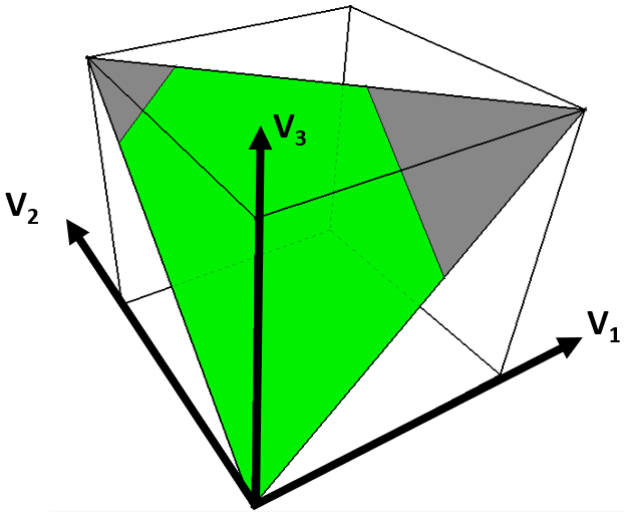


Ilustración 6 – Aplicación de restricciones de capacidad de flujo

Se cuenta, por lo tanto, con un espacio de soluciones que cumple con todas las restricciones del modelo, tanto las de balance en estado estacionario como las restricciones de capacidad de flujo. Gráficamente, este espacio de soluciones tiene un volumen de $\sqrt{7500} - \sqrt{192} - \sqrt{12}$ (o aproximadamente 69.2820323 u^2). $\sqrt{7500}$ corresponde al área del triángulo verde de la Ilustración 5, y $\sqrt{192}$ y $\sqrt{12}$ corresponden a los triángulos grises de la Ilustración 6. En este caso, el llamado "volumen" del espacio de soluciones es un área, al ser plano el espacio de soluciones.

El hecho de tener un espacio de soluciones en dos dimensiones se puede deducir a partir de la matriz estequiométrica del modelo. Dicha matriz consta de tres columnas, cada una de ellas correspondiendo a uno de los flujos, y una sola fila, correspondiendo al único metabolito del sistema. Recordando la ecuación del balance en estado estacionario:

$$SV = 0$$

donde S es la matriz estequiométrica y V es el vector de flujos metabólicos, se puede entender este sistema como un sistema de ecuaciones indeterminado, donde los flujos metabólicos son las variables. Por lo tanto, en este caso, se tienen tres variables (V_1, V_2 y V_3) y una sola ecuación (que indica que la suma de V_1 y V_2 tiene que ser igual a V_3), por lo que dos variables quedan libres y la tercera se puede determinar a partir de ellas. Esto lleva a que el espacio de soluciones conste de dos dimensiones, a pesar de que haya tres flujos en el sistema.

Cabe destacar que, si bien estas consideraciones se deducen a partir de este modelo ramificado simple, las mismas se pueden extrapolar a los modelos metabólicos más complejos. En todos los casos, la cantidad de metabolitos va a ser inferior a la cantidad de flujos metabólicos, por lo que en general el sistema va a ser indeterminado y se va a contar con un espacio de soluciones de menos dimensiones que la cantidad total de

flujos del sistema (la única alternativa es que el sistema sea incompatible, en cuyo caso el volumen es cero). También es importante tener en cuenta las conservaciones que puedan aparecer en el modelo, es decir, filas de la matriz estequiométrica que sean linealmente dependientes y por lo tanto no aporten restricciones adicionales, lo que aumenta la dimensionalidad del espacio de soluciones, según se verá más adelante.

Es necesario tener en cuenta las dificultades que se presentan a la hora de hacer un muestreo en el espacio de soluciones al estar este comprendido dentro de un espacio de más dimensiones. En el caso del modelo ramificado simple, por ejemplo, se desea tomar muestras en un plano que está descrito dentro de un espacio tridimensional. Si uno decide sortear valores de V_1 y V_2, y calcular V_3 a partir de ellos, estaría viendo solamente una proyección del espacio de soluciones al plano determinado por los ejes de V_1 y V_2, y no el espacio de soluciones original. Lo mismo sucede al sortear valores de V_1 y V_3 o V_2 y V_3. La Ilustración 7 muestra estas proyecciones del espacio de soluciones, a modo de presentar esta dificultad en forma gráfica. Por este motivo, es necesario generar un espacio de referencia adecuado para llevar a cabo los muestreos, según se detalla más adelante en la metodología, en la sección 4.4. En el caso del modelo ramificado simple, el espacio de referencia deberá ser bidimensional y estar ubicado en el mismo plano en que se ubica el espacio de soluciones.

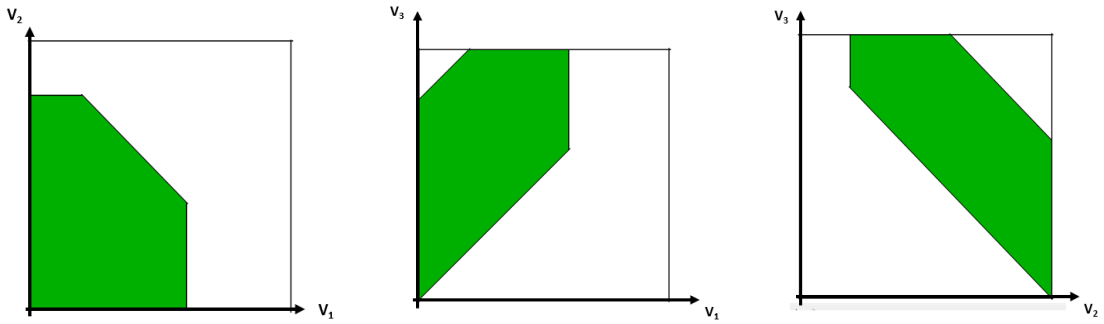


Ilustración 7 – Proyecciones del espacio de soluciones del modelo ramificado simple

4.1.2. Modelo núcleo de E.coli (E.coli core model)

El modelo núcleo de E.coli [5] es un modelo basado en restricciones simplificado, aunque contiene suficiente información del metabolismo de la bacteria como para llevar a cabo cálculos significativos. La representación matricial de este modelo consiste en una matriz estequiométrica de 72 filas (que corresponden a 72 metabolitos) por 95 columnas (que corresponden a 95 reacciones).

En el modelo hay 20 reacciones de intercambio (una para cada metabolito extracelular). También hay 25 reacciones de transporte, 49 reacciones metabólicas y una reacción de producción de biomasa. Los metabolitos están también clasificados en extracelulares (para un total de 20) y citosólicos (para un total de 52). De estos 72 metabolitos, hay 54 que son únicos, mientras que la mayoría de los metabolitos extracelulares son simplemente versiones extracelulares de los metabolitos citosólicos.

Se cuenta, además, con la clasificación de las reacciones en los siguientes sub-sistemas: reacciones anapleróticas, ciclo del ácido cítrico, intercambio, metabolismo de glutamato, glucólisis/gluconeogénesis, transporte y metabolismo de iones inorgánicos,

fosforilación oxidativa, ruta de la pentosa fosfato, metabolismo de piruvato, y transporte extracelular.

La Ilustración 8 presenta una representación del modelo. Los marcos azules representan la membrana citoplasmática (superficie interna y externa). Los metabolitos citosólicos están representados como círculos naranjas, y los metabolitos extracelulares están representados como círculos amarillos.

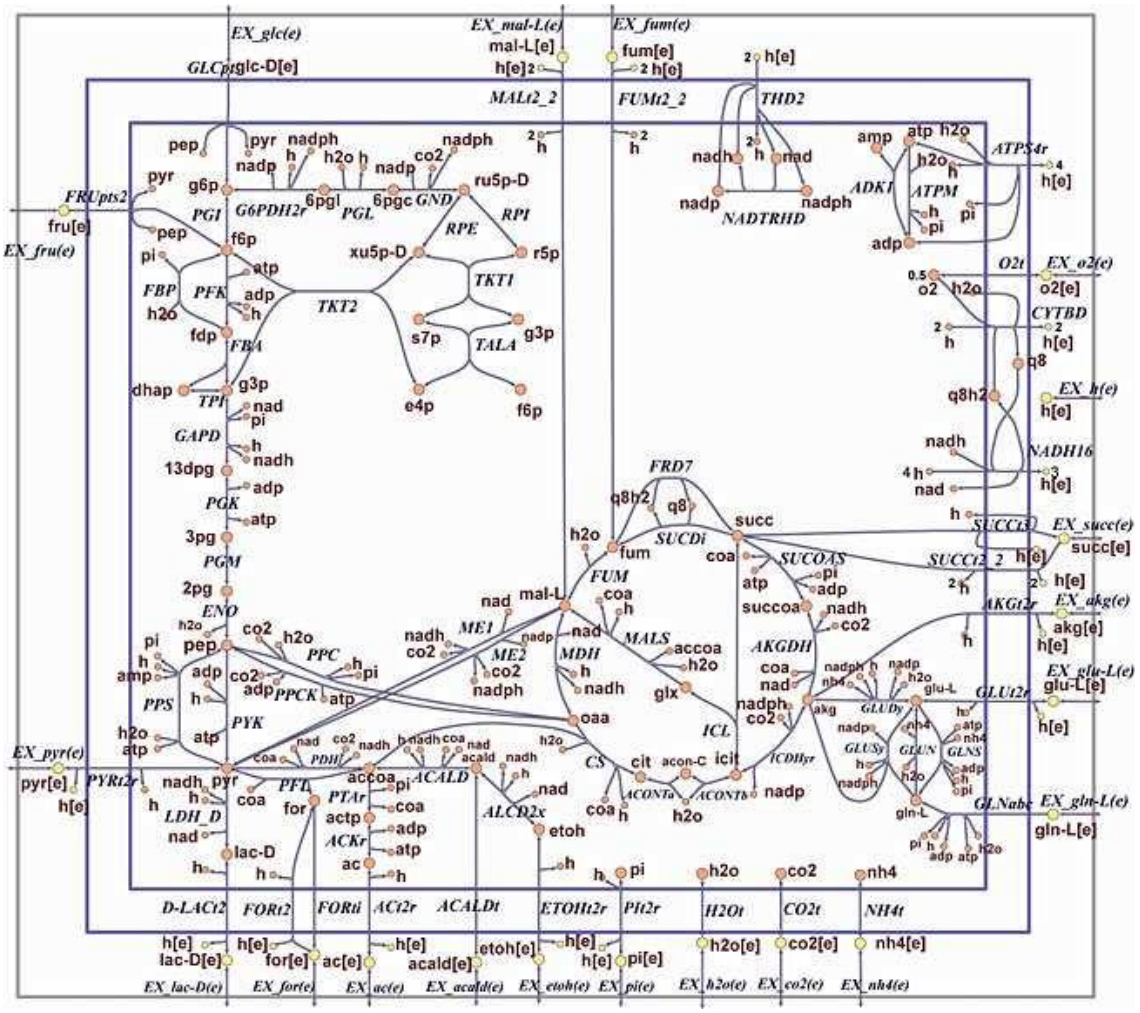


Ilustración 8 – Modelo núcleo, tomado de [5]

Las reacciones del modelo núcleo fueron seleccionadas para representar las vías metabólicas del metabolismo central de E.coli. Algunas vías, tales como la cadena de transporte de electrones, fueron simplificadas ampliamente para reducir el tamaño del modelo.

La matriz estequiométrica del modelo y las restricciones de LB y UB se pueden descargar de la siguiente dirección: http://systemsbiology.ucsd.edu/Downloads/EcoliCore [15]. En este trabajo, el modelo núcleo se toma como base para generar un modelo más simplificado a efectos de poder hacer cálculos en su espacio de soluciones utilizando métodos más directos. Hemos denominado a este modelo más simplificado "núcleo compacto de E.coli", y su construcción y validación se detalla en los resultados (5.2).

4.2. Software libre utilizado

4.2.1. GLPK

Para todos los problemas de optimización que se presentaron en el desarrollo del presente trabajo, se empleó el paquete de software GLPK (GNU Linear Programming Kit) [16]. Este paquete forma parte del proyecto GNU.

A modo de ejemplo, el siguiente pseudocódigo presenta la resolución de FVA para un modelo metabólico de m metabolitos y n reacciones en el formato usado en GLPK.

```
param S{j in 1..m, i in 1..n};

param Vmax{i in 1..n};

param Vmin{i in 1..n};

var v{i in 1..n};

for i in 1..n:

        maximize/minimize v[i]

        s.a balance{j in 1..m}: sum{i in 1..n} S[j,i]*v[i]=0

        s.a upperbound{i in 1..n}: v[i] <= Vmax[i]

        s.a lowerbound{i in 1..n}: v[i] >= Vmin[i]
```

4.2.2. COBRApy

Se hizo uso del paquete COBRApy (Constraints-Based Reconstruction and Analysis for Python) [17], diseñado para modelar y analizar redes metabólicas en Python. El paquete se puede descargar de la siguiente dirección: https://opencobra.github.io/cobrapy/. Asimismo, la documentación correspondiente se puede encontrar en: https://cobrapy.readthedocs.io/en/latest/.

En particular, el paquete cuenta con la función “model.optimize” que permite correr FBA en un modelo, y la función “flux_variability_analysis” que permite correr FVA.

4.3. Preparación de los modelos

En esta sección se detallan algunos de los procedimientos que se usan para preparar los modelos metabólicos para su estudio. En particular, se trata el tema de eliminar las filas linealmente dependientes de las matrices estequiométricas, que no aportan restricciones adicionales. También se trata el tema de cómo configurar un flujo fijo en un determinado valor en un modelo, lo cual no es trivial y resulta de utilidad para los resultados presentados en las secciones 5.1.5 y 5.4.2.

4.3.1. Eliminación de filas linealmente dependientes

Es importante tener en cuenta que las matrices estequiométricas correspondientes a redes metabólicas reales suelen presentar filas linealmente dependientes. Estas filas deben ser eliminadas si se desea trabajar con matrices de rango completo, lo cual

simplifica mucho la aplicación de algunas de las metodologías que se presentan en este trabajo, tales como la generación de una base ortonormal del espacio nulo de la matriz estequiométrica mediante su descomposición QR según lo que se presenta en la sección 4.4.1.2.1.

La existencia de filas linealmente dependientes es causada por relaciones de conservación entre determinados metabolitos [18]. Es decir, que una combinación lineal de las concentraciones de determinados metabolitos se mantiene constante en el tiempo. Un ejemplo típico es el siguiente:

$$ATP + ADP + AMP = constante$$

Esta y otras relaciones de conservación se pueden ver al analizar la matriz del modelo núcleo compacto de E.coli presentado en la sección 5.2.2, según se detallará más adelante.

Es necesario entonces contar con una forma eficiente de identificar las conservaciones en un modelo para poder eliminar las filas linealmente dependientes de su matriz estequiométrica, las cuales no aportan información adicional para los análisis correspondientes al presente trabajo. Se eligió hacer uso de la librería open source de Python "SciPy", con su sub-paquete "linalg" que contiene las funciones de álgebra lineal. La función "scipy.linalg.qr" da una forma de hacer la descomposición QR de una matriz, y permite elegir usar o no la opción de "pivoteo de columnas", la cual introduce una matriz de permutación P, donde:

$$AP = QR$$

siendo A la matriz original. La matriz de permutación da un ordenamiento de las columnas de la matriz A donde las columnas que son combinaciones lineales de otras van al final. Para hacer uso de esto en un modelo metabólico basado en restricciones, se establece que A sea la transpuesta de la matriz estequiométrica (S^T), de modo que la matriz P resultante de la descomposición QR con pivoteo de columnas devuelva un ordenamiento de las filas de la matriz estequiométrica en el que las filas que son combinaciones lineales de otras van al final.

La librería open source de Python "NumPy", en su sub-paquete "linalg", contiene la función "numpy.linalg.matrix_rank", la cual devuelve el rango de una matriz. Usando esta función, se puede calcular el rango de la matriz estequiométrica, y así saber cuántas filas se deben eliminar para tener una matriz de rango completo. Eliminando entonces las filas que la matriz P indica como combinaciones lineales, la matriz estequiométrica pasa a ser de rango completo pero sigue conteniendo toda la información necesaria para los análisis que se describen en este trabajo.

4.3.2. Metodología para fijar flujos

Existen situaciones en las que es necesario modelar una o más reacciones en un modelo metabólico con flujos fijos. Es decir, más allá de introducir límites superiores e inferiores en las restricciones de capacidad de flujos, hay veces en las que se vuelve necesario fijar un flujo a un determinado valor. Si bien en la resolución de problemas de optimización esta consideración no implica una dificultad adicional, a la hora de realizar muestreos aleatorios sí se la debe tener en cuenta. Si se lo piensa por el lado de la

generación de muestreos aleatorios dentro de un espacio de referencia, fijar un flujo (o, en su defecto, acercar mucho el límite inferior y el límite superior del mismo) puede hacer que la probabilidad de que alguno de los puntos en el espacio de referencia sea un punto válido pase a ser infinitamente pequeña.

Por este motivo, es necesario plantear una metodología a seguir cuando se desea fijar un flujo en el modelo. Se toma como referencia el apartado "Sampling an affine space" de la sección 2.3.1 de la tesis de Jan Schellenberger [19]. Aquí, el autor proporciona una forma de convertir un problema del tipo $Ax = b$ en un problema del tipo $Ay = 0$. Esto es de mucha utilidad a la hora de fijar un flujo en el modelo, ya que una posibilidad sería eliminar la columna correspondiente a dicho flujo de la matriz estequiométrica y pasar su valor al lado derecho de la ecuación, convirtiendo un problema del tipo $SV = 0$ en un problema $S^*V^* = b$, donde S^* es la matriz estequiométrica sin el flujo fijado, V^* es el vector de flujos sin el flujo fijado, y b es un vector que contiene el valor del flujo fijado en las entradas correspondientes a las filas de la matriz estequiométrica original en las que aparecía dicho flujo y cero en las que no. Teniendo el problema en la forma $S^*V^* = b$, se puede volver a un problema del tipo $S^*y = 0$ siguiendo el planteo de Schellenberger, según se detalla a continuación:

- Siendo $Ax = b$, encontrar una solución x_0 al sistema tal que $Ax_0 = b$.
- Transformar las variables a y tal que $y = x - x_0$, es decir: $x = y + x_0$.
- Se tiene entonces $A(y + x_0) = b \rightarrow Ay + Ax_0 = b \rightarrow Ay + b = b \rightarrow Ay = 0$.
- Se hace muestreo en y para luego transformar los valores a x.

El muestreo en el espacio alternativo se puede hacer con cualquiera de las metodologías planteadas en la siguiente sección.

4.4. Cálculo de volúmenes

4.4.1. Monte Carlo directo

La primera aproximación al estudio de los espacios de soluciones en estado estacionario de los modelos presentados se hace a través del método de Monte Carlo. Este método, en su versión más simple, genera muestras aleatorias distribuidas uniformemente dentro de un espacio de referencia continuo que contiene al espacio de soluciones que se desea estudiar. Cada vez que se genera una muestra, se analiza si la misma cumple con todas las restricciones que indican que esta está dentro del espacio de soluciones, o si queda fuera del mismo. Este algoritmo presenta una forma sencilla de generar muestras dentro del espacio de soluciones, pero también puede usarse para estimar el volumen del espacio de soluciones, siempre y cuando se conozca el volumen del espacio de referencia.

4.4.1.1. Esquema general

En primer lugar, se presenta el esquema general del método de Monte Carlo directo con el pseudocódigo correspondiente. Este método es la base de las demás aplicaciones metodológicas presentadas en este trabajo. El pseudocódigo que se encuentra a

continuación corresponde a la utilización del método en su versión más simple para estimar un volumen a partir de la generación de puntos aleatorios dentro de un espacio de referencia continuo.

En el siguiente algoritmo, se utiliza una muestra aleatoria $(X^{(1)}, \dots, X^{(n)})$ de tamaño n, donde cada $X^{(j)} = (X_1^{(j)}, \dots, X_m^{(j)})$ se sortea de acuerdo a una distribución uniforme en un espacio de referencia $\mathcal{K}^m$ y se intenta aproximar el volumen de una región $\mathcal{R}$.

Entrada: espacio de referencia $\mathcal{K}^m$, región $\mathcal{R}$, tamaño de la muestra n.
Salida: estimación del volumen $\bar{\lambda}(\mathcal{R})$, estimación de la varianza $V[\bar{\lambda}(\mathcal{R})]$.

1. Inicialización $S = 0$.
2. For $j = 1, \dots, n$:
   a. Sortear $X^{(j)}$ con distribución uniforme en $\mathcal{K}^m$.
   b. If $X^{(j)} \epsilon \mathcal{R}$ then $\phi(X^{(j)}) = 1$ else $\phi(X^{(j)}) = 0$.
   c. Acumular $S = S + \phi(X^{(j)})$.
3. Estimador puntual del volumen $\bar{\lambda}(\mathcal{R}) = vol(\mathcal{K}^m) * S/n$.
4. Estimador puntual de la varianza $V[\bar{\lambda}(\mathcal{R})] = vol(\mathcal{K}^m)^2 * \frac{(S/n)(1-S/n)}{n-1}$.

El pseudocódigo muestra que, para obtener el volumen del espacio de soluciones, se toma la cantidad de hits sobre la cantidad de muestras como la proporción que el espacio de soluciones ocupa en el espacio de referencia. Por lo tanto, multiplicando este estimador por el volumen del espacio de referencia, se obtiene una estimación del volumen del espacio de soluciones.

Es claro que, para poder aplicar el método Monte Carlo directo en el cálculo del volumen del espacio de soluciones, es necesario tener un espacio de referencia continuo de volumen conocido que contenga al espacio de soluciones y que se ajuste lo más posible a él, de manera de obtener una buena proporción de hits respecto a la cantidad de muestras generadas. En la siguiente sección se presentan distintas formas de construir dicho espacio de referencia para modelos metabólicos basados en restricciones.

4.4.1.2. Determinación del espacio de referencia

La determinación del espacio de referencia dentro del cual se hace el sorteo es una de las partes más difíciles de la obtención de muestras a partir del método de Monte Carlo directo. Un espacio de referencia correctamente diseñado tiene que cumplir con determinados criterios:

- Debe contener completamente al espacio de soluciones que se desea muestrear.
- Debe ajustarse lo más posible al espacio de soluciones. No puede ser tan grande que haga que ninguna de las muestras sorteadas caiga dentro del espacio de soluciones.
- Debe tener las mismas dimensiones que el espacio de soluciones. A modo de ejemplo, no se podría usar un cubo como espacio de referencia para sortear puntos dentro de un plano, ya que ninguno de los puntos muestreados caería exactamente en el plano.
- Debe ser un espacio dentro del cual se pueda hacer sorteos con facilidad.
- Idealmente, es un espacio cuyo volumen es conocido, ya que esto permite usarlo para estimar el volumen del espacio de soluciones.

Dados estos requerimientos, se presentan a continuación dos formas de obtener un espacio de referencia a la hora de aplicar el método de Monte Carlo a modelos metabólicos basados en restricciones: determinando un paralelepípedo con la misma dimensión que el espacio de soluciones y que lo contenga, o determinando una base ortonormal del espacio nulo de la matriz estequiométrica.

Se destaca que las muestras obtenidas mediante ambos métodos siempre cumplen con la restricción de $SV = 0$ (donde S es la matriz estequiométrica y V es el vector de flujos correspondiente a la muestra), porque los espacios de referencia que se generan siempre están comprendidos en el espacio nulo de la matriz estequiométrica. Por esto, para aplicar el método de Monte Carlo directo, es necesario solamente revisar en cada iteración si la muestra sorteada cumple con todas las restricciones de LB y UB, pero no es necesario revisar que cumpla con $SV = 0$.

El espacio nulo de S corresponde a todos aquellos vectores que cumplen que, si se multiplica la matriz estequiométrica por ellos, el resultado es el vector nulo. Se entiende entonces que todos los puntos del espacio nulo equivalen a configuraciones de flujo que cumplen con el estado estacionario (pero no necesariamente con las restricciones de capacidad de flujo del modelo). Tomando esto como un sistema de ecuaciones indeterminado (la cantidad de metabolitos es siempre menor que la cantidad de reacciones), se entiende que la dimensión del espacio nulo corresponde a la diferencia entre la cantidad de reacciones y la cantidad de metabolitos, correspondientes a filas linealmente independientes. Hay que tener en cuenta que puede haber filas de la matriz estequiométrica que sean linealmente dependientes, como se menciona en la sección 4.3.1.

4.4.1.2.1. Determinación del espacio de referencia mediante una base ortonormal

Una forma de determinar el espacio de referencia consiste en definir una base ortonormal del espacio nulo de la matriz estequiométrica, que es donde se encuentra contenido el espacio de soluciones, y utilizar esa base ortonormal para construir el espacio de referencia.

4.4.1.2.1.1. Descomposición QR

El primer paso consiste en hacer la descomposición QR de la transpuesta de la matriz estequiométrica del modelo metabólico. La descomposición QR consiste en la descomposición de una matriz en dos matrices: una matriz ortogonal Q y una matriz triangular superior R. A partir de las últimas columnas de la matriz Q, se obtiene una base ortonormal del espacio nulo de la matriz estequiométrica. La cantidad de columnas de la matriz Q que se deben tomar corresponde a la dimensión del espacio nulo de la matriz estequiométrica. Para conocer la dimensión de dicho espacio nulo, como se mencionó, alcanza con conocer la cantidad de metabolitos independientes y de reacciones en el modelo.

Las últimas columnas de la matriz Q, entonces, conforman la matriz W, cuyas columnas corresponden a la base ortonormal del espacio nulo de S. Cabe destacar que las columnas de W, por ser esta una base ortonormal, son vectores unitarios y perpendiculares entre sí. Multiplicando los vectores de esta base por escalares aleatorios, se obtienen puntos que cumplen con las restricciones del balance de masa en

estado estacionario, y puede posteriormente controlarse si estos puntos cumplen o no con las restricciones de capacidad de flujo de cada reacción (LB y UB).

4.4.1.2.1.2. Optimización de los rangos

Una vez obtenida la matriz W, es posible generar puntos dentro del espacio nulo de la matriz estequiométrica, lo que garantiza que estos puntos cumplan con la condición de estado estacionario del sistema. Sin embargo, el espacio de referencia deseado, si bien está dentro de este espacio nulo, todavía no está completamente definido. Para terminar de definirlo, es necesario encontrar los rangos dentro de los cuales se deben sortear los escalares aleatorios que van a multiplicar a los vectores de la matriz W, de manera de delimitar una caja (un hiperrectángulo) como espacio de referencia.

En teoría, existen muchas opciones distintas para estos rangos, siempre y cuando los mismos garanticen que el espacio de soluciones quede completamente contenido en el espacio de referencia. Sin embargo, lo ideal es ajustar el espacio de referencia lo más posible al espacio de soluciones, de manera de hacer los muestreos de Monte Carlo lo más eficientes posible. Esto se consigue resolviendo problemas de optimización para cada variable aleatoria.

La Ilustración 9 presenta esta idea de forma gráfica, para un espacio nulo de dos dimensiones (como podría ser el del modelo de ramificación simple presentado en la sección 4.1.1). En la Ilustración 9, el espacio nulo se presenta como todo el plano en color azul. Dentro de él, se ubica el espacio de soluciones del modelo, pintado en rosado, que corresponde a todas aquellas configuraciones de flujo que, además de cumplir con el estado estacionario, cumplen con las restricciones adicionales de capacidad de flujo. El espacio de referencia más ajustado al espacio de soluciones es el rectángulo gris, comprendido entre x_1^{min} y x_1^{max} en una de las direcciones de la base, y entre x_2^{min} y x_2^{max} en la otra. Un punto cualquiera muestreado uniformemente se obtiene a partir del sorteo de un valor x_1 y un valor x_2, según la ecuación

$$V = WX$$

donde V es una configuración de flujos que cumple con el estado estacionario, W es la base ortonormal del espacio nulo y X es un vector aleatorio.

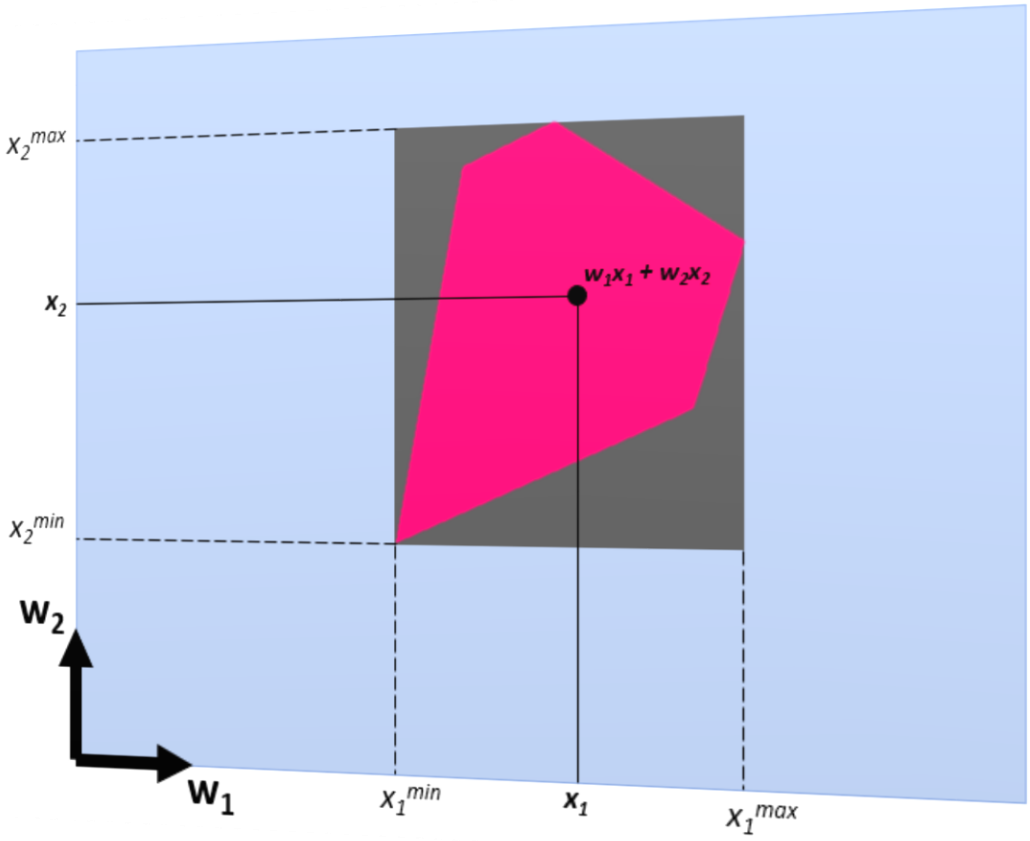


Ilustración 9 – Determinación del espacio de referencia según una base ortonormal

El siguiente paso de esta metodología consiste entonces en resolver los problemas de optimización que permitan ajustar el sorteo de cada variable aleatoria al rango óptimo. Se dan dos problemas de programación lineal por cada variable (uno de minimización y otro de maximización), en los que las restricciones implican que se cumplan las restricciones de capacidad de flujo del modelo. El siguiente pseudocódigo presenta estos problemas de optimización, donde d es la cantidad de flujos del modelo y n representa la cantidad de dimensiones del espacio nulo.

Entrada: matriz W, LB y UB de flujos.
Salida: máximo y mínimo de $x_k \; \forall \; k \; in \; (1 \dots n)$.
for $k = 1, \dots, n$:
maximizar/minimizar x_k
s.a:

$$\sum_{i=1}^{n} W_{j,i} x_i \leq ub_j \; \forall \; j \; in \; (1 \dots d)$$

$$\sum_{i=1}^{n} W_{j,i} x_i \geq lb_j \; \forall \; j \; in \; (1 \dots d)$$

La resolución de estos problemas para este trabajo se llevó a cabo usando el paquete de software GNU Linear Programming Kit (GLPK).

4.4.1.2.1.3. Volumen del espacio de referencia

El volumen del espacio de referencia es fácil de calcular, al ser conocida la base ortonormal a partir de la cual se define el mismo. Simplemente se multiplican los rangos de sorteo en cada dirección, calculando así el volumen del hiperrectángulo de referencia. Por ejemplo, para la figura de la Ilustración 9 el volumen del espacio de referencia (un área en este caso particular) sería:

$$vol = (x_1^{max} - x_1^{min})(x_2^{max} - x_2^{min})$$

4.4.1.2.2. Determinación del espacio de referencia como paralelepípedo

Otra forma de determinar el espacio de referencia se hace en base a la metodología usada por Price et al. [11]. El método consiste en usar la matriz estequiométrica para definir un paralelepípedo que tenga la misma dimensión que el espacio de soluciones y que lo contenga. El volumen del paralelepípedo se calcula por medio de la siguiente fórmula [20].

$$volumen = \sqrt{\det(B^T B)}$$

donde las columnas de la matriz B son los ejes del paralelepípedo.

El siguiente pseudocódigo indica el procedimiento a seguir para el cálculo del estimador del volumen del espacio de soluciones por este método. Es necesario aclarar que este método solo funcionaría para modelos pequeños, ya que, al aumentar el tamaño del modelo, la cantidad de hits disminuiría mucho, porque el espacio de referencia pasa a ser mucho más grande que el espacio de soluciones.

Entrada: matriz B, LB y UB de flujos, tamaño de la muestra n.
Salida: estimación del volumen $\bar{\lambda}(\mathcal{R})$, estimación de la varianza $V[\bar{\lambda}(\mathcal{R})]$.

1. Inicialización $S = 0$.
2. For $j = 1, \dots, n$:
   a. Sortear $X^{(j)}$ con distribución uniforme en $\mathcal{J}^m$.
   b. Calcular flujos $V = BX^{(j)}$.
   c. If $v_i > lb_i$ and $v_i < ub_i$ $\forall v_i \in V$ then $\phi(X^{(j)}) = 1$ else $\phi(X^{(j)}) = 0$.
   d. Acumular $S = S + \phi(X^{(j)})$.
3. Estimador puntual del volumen $\bar{\lambda}(\mathcal{R}) = \sqrt{\det(B^T B)} * S/n$.
4. Estimador puntual de la varianza $V[\bar{\lambda}(\mathcal{R})] = \det(B^T B) * \frac{(S/n)(1-S/n)}{n-1}$.

4.4.2. Muestreo por importancia (importance sampling)

4.4.2.1. Introducción

Se propone el uso de la técnica de muestreo por importancia como posible forma de mejorar el muestreo conseguido mediante Monte Carlo directo para el espacio de soluciones de un modelo metabólico. Teniendo en cuenta que el volumen de dicho espacio es muy pequeño en relación al volumen de la caja de muestreo que se puede encontrar mediante los métodos presentados, se entiende que el uso de muestreo por importancia podría ser de utilidad para encontrar muestras válidas en casos de modelos con espacios de soluciones en muchas dimensiones, en los que un método Monte Carlo directo requiera una cantidad excesiva de iteraciones para generar muestras válidas.

La base de este método está en darle más importancia al muestreo sobre la región de interés (en este caso, el espacio de soluciones). Esto se consigue mediante el uso de una

distribución no uniforme para el muestreo, y el posterior ajuste del estimador para tener en cuenta el método usado.

Si bien el muestreo por importancia puede traer grandes ganancias a la hora de resolver un problema mediante Monte Carlo, también puede pasar lo contrario: el muestreo por importancia puede devolver un estimador con varianza infinita cuando el uso de Monte Carlo directo hubiera dado una varianza finita. Se reconoce como el método de reducción de varianza más difícil de implementar de forma correcta [21].

4.4.2.2. Cálculo del estimador

Según se mencionó, es necesario ajustar la forma de calcular el estimador para usar muestreo por importancia en Monte Carlo. Para el método Monte Carlo directo, el estimador empieza en 0, y se le va sumando 1 cada vez que una muestra cae dentro del espacio de soluciones. Al completarse todas las iteraciones, el estimador indica cuántas de ellas produjeron muestras dentro del espacio de soluciones, permitiendo así obtener la fracción que el espacio de soluciones ocupa dentro del espacio de muestreo. En el caso de muestreo por importancia, este procedimiento no daría una solución correcta, ya que las muestras tienen mayor probabilidad de caer en una determinada región y menor probabilidad de caer en otra, por lo que es necesario "corregir" el estimador.

Para esto, se calcula un coeficiente de verosimilitud $L(x)$ para cada muestra, que se define de la siguiente forma:

$$L(x) = \frac{f_{distr.original}(x)}{f_{distr.nueva}(x)}$$

Donde $f_{distr.original}$ corresponde a la función de densidad de probabilidad de la distribución uniforme, y $f_{distr.nueva}$ corresponde a la función de densidad de probabilidad de la distribución elegida para el muestreo por importancia (triangular o exponencial truncada según se detalla en la sección 4.4.2.4).

En el caso de la distribución uniforme, se tiene que:

$$f(x) = \frac{1}{b-a}$$

para cualquier x comprendido en el intervalo $[a, b]$.

En el caso de la distribución triangular y la distribución exponencial truncada, sus funciones de densidad de probabilidad se detallan en la sección 4.4.2.4.

Es importante notar que, cuando el sorteo para Monte Carlo es en más de una dimensión, el coeficiente de verosimilitud tiene que ser calculado por separado para cada dimensión, para luego calcular un coeficiente de verosimilitud general para la muestra que surge de la productoria de los coeficientes de verosimilitud de cada dimensión. Es decir, para un caso de espacio de referencia en m dimensiones,

$$L(x) = \prod_{i=1}^{m} L(x)_i$$

En el cálculo de cada $L(x)_i$, al calcular $f(x)$ para la distribución uniforme, deben tomarse los límites de la caja de muestreo en la dimensión correspondiente como los límites del intervalo en el que se hace la distribución uniforme.

Finalmente, el estimador S se actualiza acumulando en cada iteración j de la siguiente manera:

$$S_j = S_{j-1} + L(x)_j$$

En un caso de Monte Carlo con n iteraciones, el estimador puntual del volumen se calcula de la siguiente forma:

$$\bar{\lambda}(\mathcal{R}) = vol(\mathcal{K}^m) * S/n$$

donde $\mathcal{K}^m$ es la caja de muestreo. Cabe destacar que esto también aplica a un método Monte Carlo directo, donde tanto la distribución original como la nueva distribución corresponden a la distribución uniforme, y por lo tanto $L(x)$ tomaría siempre un valor de 1.

4.4.2.3. Varianza

El cálculo de la varianza, según se lo plantea a continuación, requiere acumular la suma de los cuadrados de $L(x)$ en cada iteración. Para obtener esta suma, se inicializa una variable que se va actualizando en cada iteración, sumándole el cuadrado correspondiente. Posteriormente, se calcula la varianza de la siguiente forma [22]:

$$V\left[\bar{\lambda}(\mathcal{R})\right] = vol(\mathcal{K}^m)^2 * \frac{\left(\frac{SC}{n-1}\right) - \left(\frac{n(S/n)^2}{n-1}\right)}{n}$$

donde SC es la suma de los cuadrados: $SC = \sum_{j=1}^{n}[L(x)_j]^2$. Cabe destacar que esta forma de calcular la varianza también aplica al método Monte Carlo directo cuyo pseudocódigo se presentó en la sección 4.4.1.1, donde $L(x)_j = 1 \; \forall j$ y por lo tanto $SC = S$, pudiendo simplificar la ecuación planteada para la varianza para llevarla a la ecuación presentada en dicho pseudocódigo.

4.4.2.4. Distribuciones

A continuación se presentan las dos distribuciones que se probaron en este trabajo: distribución triangular y distribución exponencial truncada. Ambas requieren, además de los extremos de la caja de muestreo, las coordenadas de un punto que se ubique dentro del espacio de soluciones. A partir de estos datos, pueden generarse distribuciones en cada dimensión de la base ortonormal del espacio que permitan muestrear con mayor probabilidad en los alrededores de dicho punto, logrando obtener muestras válidas incluso cuando esto no sería probable para un método Monte Carlo directo.

4.4.2.4.1. Distribución triangular

Se trata de una distribución de probabilidad continua con parámetros *a*, *b* y *c*, donde *a* y *b* representan los extremos, y *c* representa un punto entre los dos extremos donde se intensifica la densidad de probabilidad. Su función de densidad es:

$$f(x) = P(x) = \begin{cases} 0\ para\ x < a \\ \dfrac{2(x-a)}{(b-a)(c-a)}\ para\ a \le x < c \\ \dfrac{2}{b-a}\ para\ x = c \\ \dfrac{2(b-x)}{(b-a)(b-c)}\ para\ c < x \le b \\ 0\ para\ b < x \end{cases}$$

La Ilustración 10 presenta dicha función de densidad.

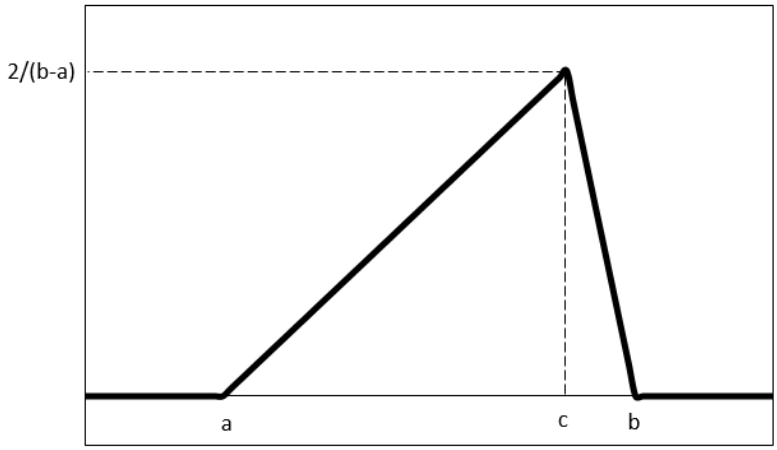


Ilustración 10 – Función de densidad de la distribución triangular

Tomando como *a* y *b* los extremos de la caja de muestreo en una determinada dirección, y como *c* la coordenada en esa dimensión de un punto que se ubique dentro del espacio de muestreo, pueden generarse muestras más concentradas alrededor de dicho punto. Dependiendo de cómo esté elegido ese punto, este procedimiento puede aumentar las probabilidades de obtener muestras dentro del espacio de soluciones para modelos en muchas dimensiones.

La función de distribución acumulada es la siguiente:

$$F(x) = P(X \le x) \begin{cases} 0\ para\ x \le a \\ \dfrac{(x-a)^2}{(b-a)(c-a)}\ para\ a < x \le c \\ 1 - \dfrac{(b-x)^2}{(b-a)(b-c)}\ para\ c < x < b \\ 1\ para\ b \le x \end{cases}$$

4.4.2.4.1.1. Sorteo de variables a partir de la distribución uniforme

A partir de la distribución acumulada, se deriva la inversa $F^{-1}(x)$:

$$x = F(F^{-1}(x))$$

Para $a < F^{-1}(x) \le c$:

$$x = \frac{(F^{-1}(x) - a)^2}{(b-a)(c-a)}$$

$$F^{-1}(x) = a + \sqrt{x(b-a)(c-a)}$$

$a < F^{-1}(x) \leq c$ implica:

- $a < a + \sqrt{x(b-a)(c-a)} \rightarrow \sqrt{x(b-a)(c-a)} > 0 \rightarrow x > 0$
- $a + \sqrt{x(b-a)(c-a)} \leq c \rightarrow x(b-a)(c-a) \leq (c-a)^2 \rightarrow x \leq \frac{c-a}{b-a}$

Para $c < F^{-1}(x) < b$:

$$x = 1 - \frac{(b - F^{-1}(x))^2}{(b-a)(b-c)}$$

$$1 - x = \frac{(b - F^{-1}(x))^2}{(b-a)(b-c)}$$

$$F^{-1}(x) = b - \sqrt{(1-x)(b-a)(b-c)}$$

$c < F^{-1}(x) < b$ implica:

- $b - \sqrt{(1-x)(b-a)(b-c)} < b \rightarrow \sqrt{(1-x)(b-a)(b-c)} > 0 \rightarrow (1-x) > 0 \rightarrow x < 1$
- $c < b - \sqrt{(1-x)(b-a)(b-c)} \rightarrow (1-x)(b-a)(b-c) < (b-c)^2 \rightarrow 1 - x < \frac{b-c}{b-a} \rightarrow x > \frac{b-a-b+c}{b-a} \rightarrow x > \frac{c-a}{b-a}$

En resumen, y teniendo en cuenta que $F(c) = \frac{(c-a)^2}{(b-a)(c-a)} = \frac{c-a}{b-a}$, se pueden generar variables aleatorias distribuidas triangularmente con el siguiente procedimiento, según los parámetros *a*, *b* y *c*:

- Se genera una variable *U* distribuida uniformemente en el intervalo (0,1)
- Se calcula *X* tal que:

$$X = \begin{cases} a + \sqrt{U(b-a)(c-a)}\ para\ 0 < U \leq F(c) \\ b - \sqrt{(1-U)(b-a)(b-c)}\ para\ F(c) < U < 1 \end{cases}$$

La variable *X* tiene una distribución triangular con parámetros *a*, *b* y *c*.

4.4.2.4.1.2. Cálculo de L(x)

Para el cálculo del estimador, es necesario encontrar $f(x)$ en cada iteración y para cada dimensión. Una vez generada una muestra en una dimensión, se verifica si es menor o igual a *c* o mayor a *c*, y se aplica la función de densidad de probabilidad (PDF) según el caso que corresponda, teniendo en cuenta que *c* representa la coordenada del punto elegido como punto donde se concentra el muestreo en esa dimensión.

4.4.2.4.2. Distribución exponencial truncada

Se trata de una distribución de probabilidad continua con parámetro $\lambda > 0$ y truncada en *B*, cuya función de densidad es:

$$f(x) = P(x) = \frac{\lambda e^{-\lambda x}}{1 - e^{-\lambda B}}$$

La Ilustración 11 presenta dicha función de densidad para distintos valores de λ con $B = 4$.

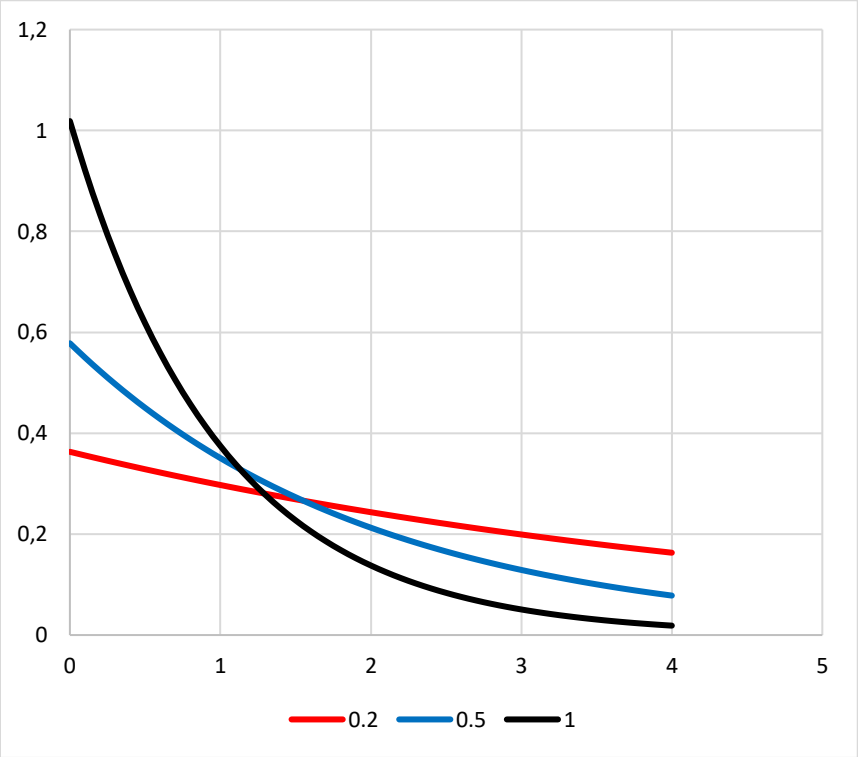


Ilustración 11 – Función de densidad de la distribución exponencial truncada

El parámetro λ modifica la probabilidad de muestreo en una región cercana al 0: cuanto mayor λ, mayor es la probabilidad de muestrear un punto cercano al 0. El parámetro B marca el punto a partir del cual la probabilidad pasa a ser 0. Para usar esta distribución en el muestreo de Monte Carlo para estimar el volumen del espacio de soluciones de un modelo, se puede hacer uso de dos distribuciones exponenciales truncadas, según el siguiente procedimiento:

- Al igual que para la distribución triangular, se selecciona un punto interior al espacio de soluciones alrededor del cual se va a concentrar el muestreo.
- En cada dimensión del muestreo, se toma la coordenada de ese punto como el "0" de la distribución exponencial. Se hace un sorteo para determinar si la muestra va a caer a la derecha o a la izquierda de dicha coordenada.
    - Para la derecha, se hace una distribución exponencial truncada con B igual al valor absoluto del máximo de la caja de muestreo en esa dimensión menos la coordenada del punto interior.
    - Para la izquierda, se hace una distribución exponencial truncada con B igual al valor absoluto de la coordenada del punto interior menos el mínimo de la caja de muestreo en esa dimensión.

El resultado de esto es una serie de muestras concentradas alrededor del punto interior seleccionado. El parámetro λ permite determinar el grado de concentración:

- Para un λ extremadamente alto, la mayoría de las muestras estarán muy cercanas al punto interior, lo que puede ayudar a obtener una mayor cantidad de muestras dentro del espacio de soluciones pero sin recorrer la totalidad del espacio.
- Para un λ extremadamente bajo (cercano a 0), el muestreo se asemeja a un muestreo uniforme y los resultados serán similares a los de un Monte Carlo directo.

Su función de distribución acumulada es:

$$F(x) = P(X \leq x) = \frac{1 - e^{-\lambda x}}{1 - e^{-\lambda B}}$$

4.4.2.4.2.1. Sorteo de variables a partir de la distribución uniforme

A partir de la distribución acumulada, se deriva la inversa:

$$x = F(F^{-1}(x))$$

$$x = \frac{1 - e^{-\lambda F^{-1}(x)}}{1 - e^{-\lambda B}}$$

$$1 - e^{-\lambda F^{-1}(x)} = x(1 - e^{-\lambda B})$$

$$e^{-\lambda F^{-1}(x)} = 1 - x(1 - e^{-\lambda B})$$

$$-\lambda F^{-1}(x) = \ln\,(1 - x(1 - e^{-\lambda B}))$$

$$F^{-1}(x) = -\frac{1}{\lambda}\ln\,(1 - x(1 - e^{-\lambda B}))$$

$0 < F^{-1}(x) < B$ implica:

- $0 < -\frac{1}{\lambda}\ln\left(1 - x\left(1 - e^{-\lambda B}\right)\right) \rightarrow \ln\left(1 - x\left(1 - e^{-\lambda B}\right)\right) < 0 \rightarrow 1 - x\left(1 - e^{-\lambda B}\right) < 1 \rightarrow\ x\left(1 - e^{-\lambda B}\right) > 0 \rightarrow x > 0$
- $-\frac{1}{\lambda}\ln\left(1 - x\left(1 - e^{-\lambda B}\right)\right) < B \rightarrow \ln\left(1 - x\left(1 - e^{-\lambda B}\right)\right) > -\lambda B \rightarrow 1 - x\left(1 - e^{-\lambda B}\right) > e^{-\lambda B} \rightarrow x < 1$

Sorteando variables con una distribución uniforme en el intervalo (0,1) y aplicándoles la inversa, se obtienen variables distribuidas según la exponencial truncada.

4.4.2.4.2.2. Cálculo de L(x)

Para el cálculo del estimador, es necesario encontrar $f(x)$ en cada iteración y para cada dimensión. Una vez generada una muestra en una dimensión, se aplica la PDF para el valor de *B* correspondiente. Asimismo, el valor obtenido debe multiplicarse por ½, para tener en cuenta el sorteo inicial que determina si la coordenada de la muestra se ubicará hacia la derecha o izquierda de la coordenada del punto interior.

4.4.3. Hit-and-Run

4.4.3.1. Introducción

Como se mencionó previamente, el método de Monte Carlo puede no ser suficiente para conseguir un muestreo efectivo del espacio de soluciones en estado estacionario de

modelos en muchas dimensiones, dada la dificultad de encontrar un espacio de referencia lo suficientemente ajustado al espacio de soluciones como para obtener una buena cantidad de muestras válidas. Por este motivo, se exploraron otras alternativas mediante el paquete COBRApy [17]. Este paquete contiene múltiples herramientas para el análisis de modelos metabólicos y procesos biológicos complejos, incluyendo una herramienta para el muestreo de flujos metabólicos, "flux sampling", que fue la que se exploró en esta parte del trabajo.

Esta herramienta permite el uso de dos métodos de muestreo: Artificial Centering Hit-and-Run (ACHR) y el algoritmo optGpSampler [23], el cual está basado en ACHR. El funcionamiento de estos algoritmos se presenta a continuación.

Para comenzar, es necesario describir el algoritmo Hit-and-Run (HR), el cual es la base de los algoritmos incluidos en la herramienta "flux sampling" del paquete COBRApy. El algoritmo HR permite evitar el problema de un espacio de referencia que no se ajuste lo suficiente al espacio de soluciones, ya que solo toma muestras válidas (que caen dentro del espacio de soluciones). Esto se consigue partiendo de un punto x_0 dentro del espacio de soluciones, y eligiendo una dirección arbitraria μ_1 en la superficie de la esfera unitaria en $\mathcal{R}^n$. La distancia de x_0 al borde del espacio de soluciones en la dirección μ_1 determina la máxima distancia que se puede recorrer en esa dirección. Se elige aleatoriamente un factor λ_1 al cual se multiplica por esa distancia máxima para obtener la distancia que se recorre entre x_0 y el siguiente punto, x_1, en la dirección μ_1. Se repite este proceso hasta obtener una cantidad satisfactoria de muestras. La Ilustración 12 muestra cómo se lleva a cabo el algoritmo.

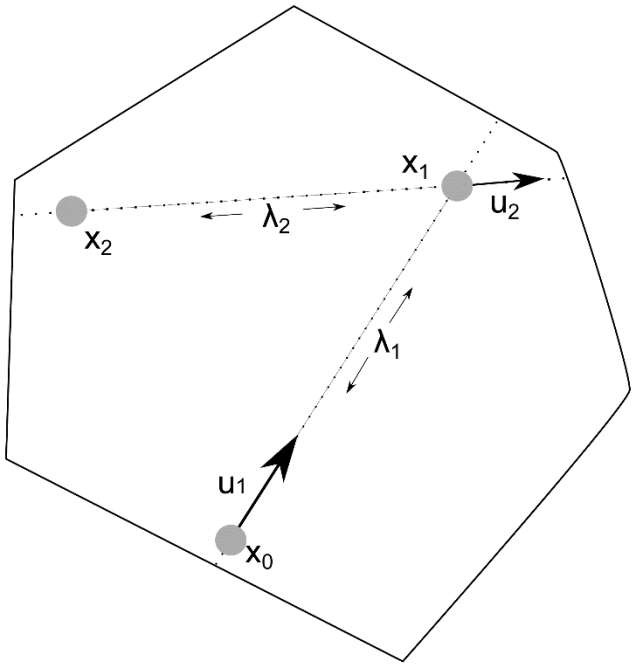


Ilustración 12 – Algoritmo HR, tomado de [23]

El hecho de usar solo el punto anterior para generar el siguiente hace que este algoritmo sea un método Markov Chain Monte Carlo (MCMC), lo que garantiza que va a converger a una distribución uniforme con elección adecuada de λ.

El algoritmo ACHR, de acuerdo a Megchelenbrink et al. [23], surge a partir de la identificación por parte de los autores de un problema al que se enfrenta comúnmente el algoritmo HR al ser aplicado al muestreo de los espacios de soluciones en estado estacionario de redes metabólicas. Se entiende que las restricciones de flujos en una red metabólica generan un espacio de flujos que es convexo pero con una forma muy irregular: es alargado para las reacciones que no están muy restringidas, y angosto para los flujos más restringidos. Esto provoca que una gran parte del espacio de soluciones se halle cercana al borde del mismo, lo cual puede significar un problema para el algoritmo HR. Ya que los puntos muestreados están cerca del borde, la distancia que se puede recorrer a partir de los mismos en la mayoría de las direcciones tiende a ser muy pequeña, provocando que todos los puntos estén muy cercanos entre sí, y dificultando la generación de un buen muestreo distribuido uniformemente en el espacio de soluciones. El algoritmo ACHR, entonces, surge como respuesta a esta problemática, al intentar hacer los muestreos en las direcciones más largas.

El algoritmo ACHR consta de dos fases: la fase de calentamiento (warm-up) y la fase principal. Durante la fase de calentamiento, se genera una serie de muestras mediante el algoritmo HR. Durante la fase principal, en cada iteración se usan todos los puntos muestreados hasta el momento para calcular un “centro” del espacio de soluciones, y se toma una dirección de muestreo que va desde uno de los puntos muestreados (seleccionado de forma aleatoria) hasta este “centro”. El hecho de que esta dirección dependa de todos los puntos muestreados hasta el momento hace que el algoritmo ACHR no corresponda a una cadena de Markov, por lo que los autores no pueden garantizar la convergencia a una distribución uniforme.

El algoritmo gpSampler hace uso de ACHR para generar muestras. Este algoritmo parte de una serie de puntos que se ubican en el borde del espacio de soluciones, generados mediante la resolución de problemas de programación lineal. A estos puntos se les aplica una transformación para llevarlos hacia el interior del espacio, y cada uno de ellos da origen a una cadena de iteraciones de ACHR, de largo determinado por el usuario. El punto final de cada cadena es lo que el algoritmo devuelve como muestreo.

Finalmente, el algoritmo optGpSampler funciona de manera similar, pero explotando una determinada cantidad de procesadores al correr esa cantidad de cadenas en paralelo. El pseudocódigo para este algoritmo se presenta en la Ilustración 13. Este algoritmo también permite al usuario determinar cada cuántos puntos generados se desea tomar una muestra.

Input: S: the solution space; T: sample count; k: step count, p: number of processors;
output: P: sequence of T sampled points;
/* Warm-up phase */
Generate $2n$ warm-up points as in part (1) of *gpSampler*'s warm-up phase;
/* Sampling-phase */
$\mathbf{P} = <>$;
$L = \lceil Tk/p \rceil$;
for $i = 1$ **to** p **do**
  $\vec{x}_0 =$ a point randomly chosen from the $2n$ warm-up points;
  for $j = 1$ **to** L **do**
    $\vec{x}_j =$ point generated from $\vec{x}_{j-1}$ by performing one iterate of the ACHR sampling phase;
    if j mod $k = = 0$ **then**
      $\mathbf{P} = <\mathbf{P}, \vec{x}_j>$;
    end
  end
end

optGpSampler(**S**, T, k, p).
doi:10.1371/journal.pone.0086587.t001

Ilustración 13 – Pseudocódigo optGpSampler, tomado de [23]

4.4.3.2. Algoritmo de cálculo de volúmenes

Para modelos con espacios de soluciones de bajas dimensiones, de acuerdo a lo presentado en el estado del arte (ver sección 2.3.3), es posible usar el método de Monte Carlo directo para generar muestras. Sin embargo, para modelos más grandes, esto se vuelve complicado, como ya se presentó, debido a la dificultad de formular un espacio de referencia que cumpla con todos los requerimientos y que a su vez se ajuste lo suficiente al espacio de soluciones como para permitir la obtención de una cantidad satisfactoria de puntos adentro del mismo. Por este motivo, se plantea la idea de usar la herramienta de muestreo de flujo (flux sampling) de COBRApy para el cálculo de los volúmenes de los espacios de soluciones en estado estacionario de redes metabólicas. Esta herramienta incluye los distintos métodos de muestreo presentados, basados en un algoritmo del tipo HR, que solo generan puntos adentro del espacio de soluciones. Por esto, es necesario desarrollar una metodología que permita usar las capacidades del sampler en el cálculo de volúmenes, ya que esto no puede hacerse de forma trivial calculando una hit fraction como en el caso de los muestreos por Monte Carlo directo en un espacio de referencia de volumen conocido.

La metodología que se propone en este trabajo está basada en la determinación de un espacio de referencia mediante una base ortonormal del espacio de soluciones que se usó en los muestreos por Monte Carlo, según lo presentado en la sección 4.4.1.2.1. Para un determinado modelo, la idea consiste en generar un modelo alternativo en COBRApy cuyo espacio de soluciones sea una caja equivalente a la que se obtendría para el modelo original si se buscara en él un espacio de referencia con la metodología presentada en la sección 4.4.1.2.1. Es decir, se busca un modelo cuyo espacio de soluciones coincida con la "caja" de muestreo del modelo original.

Una vez obtenido este modelo alternativo, las restricciones de capacidad de flujo del modelo original se van agregando una a una al modelo alternativo en COBRApy, y se van generando muestreos en cada iteración. En la última iteración, se da la última restricción de capacidad de flujo y el muestreo estaría entonces dentro del espacio de soluciones del modelo original (que está contenido en la "caja" inicial y a su vez restringido por todas las restricciones de capacidad de flujo). Es necesario destacar que las muestras generadas en cada iteración siempre cumplen con las restricciones estequiométricas del modelo original, porque se encuentran dentro de un espacio que

está generado a partir de la misma base ortonormal que el espacio de soluciones del modelo original.

Teniendo la serie de muestreos, y conociendo el volumen de la "caja" de la que se parte, puede calcularse el estimador del volumen del espacio de soluciones. Alcanza con ver, en cada iteración, qué fracción de los puntos muestreados cumple con la siguiente restricción de capacidad de flujo, y al final multiplicar todas estas fracciones por el volumen de la caja de la que se parte.

La construcción del modelo alternativo se hace a partir del modelo original, extendiendo su matriz estequiométrica para agregar la definición de las variables aleatorias *x*, según se explica a continuación. La matriz estequiométrica, al estar la red metabólica en estado estacionario, define un sistema de ecuaciones según lo que se presentó en la sección 0. La idea para la construcción del modelo alternativo es usar esto para definir las variables aleatorias *x* como parte de la matriz estequiométrica, y así poder asignar a estas variables límites superiores e inferiores en el modelo que definan la caja deseada. Como matriz estequiométrica del modelo alternativo, se plantea la matriz que se presenta en la Ilustración 14.

	V (flujos)	X (variables aleatorias)
Metabolitos originales	S (matriz estequiométrica original)	0 (matriz nula)
Metabolitos auxiliares	Diagonal (matriz diagonal -1)	W (base ortonormal)

Ilustración 14 – Matriz estequiométrica del modelo alternativo

Al generar muestras para un modelo con esta matriz estequiométrica, el sampler de COBRApy va a generar, en cada muestra, valores para los flujos originales *V* y también valores para las variables aleatorias *X*. Las primeras filas de la matriz, correspondientes a los metabolitos en el modelo original, aseguran que las muestras cumplan con el estado estacionario. Las últimas filas de la matriz, correspondientes a metabolitos auxiliares que se definen especialmente para el modelo alternativo, son las que definen las variables *X*. Si se mira una de estas filas en particular, se puede ver que ella implica:

$$-v_i + w_{i1}x_1 + \cdots + w_{in}x_n = 0$$

donde *i* es la fila en cuestión y *n* es la dimensión del espacio nulo de la matriz estequiométrica original. Esto implica que:

$$v_i = w_{i1}x_1 + \cdots + w_{in}x_n$$

lo cual define la relación entre las variables aleatorias y los flujos originales.

Respecto a las restricciones de capacidad de flujo del modelo alternativo, se les dan a las variables aleatorias *X* las restricciones que las hacen caer dentro del rango optimizado con GLPK que se plantea en la sección 4.4.1.2.1.2. A los flujos originales se les van dando los límites superiores e inferiores originales de a uno, a medida que se van haciendo muestreos sucesivos.

Este algoritmo está basado en el método presentado en las notas de la Dra. Shuchi Chawla de la Universidad de Wisconsin-Madison para el curso "CS880: Approximation Algorithms" [24]. En estas notas, la Dra. Chawla discute aplicaciones de técnicas de muestreo a la estimación de un volumen convexo en R^n. Para ello, sugiere partir de una esfera de volumen conocido que contiene al volumen a estimar, e irse acercando a este volumen reduciendo sucesivamente el tamaño de la esfera sin generar cambios demasiado grandes en su tamaño. Haciendo muestreos con un algoritmo MCMC, la autora calcula la proporción de un volumen respecto al siguiente, al igual que se hace en la metodología presentada en este trabajo.

5. Resultados

En este capítulo se presentan los principales resultados de este trabajo. En la primera sección se tratan todos los resultados obtenidos a partir del modelo ramificado simple, que sirven como base para comprender algunas de las metodologías aplicadas luego a un modelo más complejo (modelo núcleo compacto de E.coli). En la segunda sección se describe la construcción y validación de este modelo más complejo a partir del modelo núcleo de E.coli. En la tercera sección se presentan los cálculos de volumen llevados a cabo en el modelo núcleo compacto, comparando el uso de muestreos por importancia y Hit-and-Run con un método de Monte Carlo directo. Finalmente, en la cuarta sección, se presenta un análisis del modelo núcleo compacto desde un punto de vista biológico, efectuando distintos cálculos mediante la aplicación del método de Monte Carlo al modelo para obtener conclusiones.

5.1. Volumen en modelo ramificado simple

Las siguientes secciones presentan el cálculo del volumen del espacio de soluciones del modelo ramificado simple mediante distintos métodos. El objetivo de esto es ilustrar las distintas metodologías que se usan en este trabajo con un ejemplo sencillo, que puede ser visualizado en forma gráfica según se presentó en la sección 4.1.1.2. Asimismo, en este caso es posible hacer el cálculo exacto del volumen, lo que sirve de referencia para validar las metodologías.

5.1.1. Monte Carlo directo con base ortonormal

En esta sección se usa la metodología presentada en la sección 4.4.1.2.1 para la conformación del espacio de referencia del muestreo según la generación de una base ortonormal del espacio de soluciones. Posteriormente, se procede a hacer un muestreo de Monte Carlo directo en dicho espacio de referencia para calcular el volumen del espacio de soluciones. Primeramente, se parte de la matriz estequiométrica presentada en la sección 4.1.1 y se hace la descomposición QR de su transpuesta en Python usando los paquetes numPy y sciPy:

```
St=numpy.matrix.transpose(S)

Q,R=scipy.linalg.qr(St)
```

A partir de esta operación, se obtienen las siguientes matrices *Q* y *R*:

Tabla 3 – Descomposición QR para modelo ramificado simple

Q			R
-0.5774	-0.5774	0.5774	1.7321
-0.5774	0.7887	0.2113	0
0.5774	0.2113	0.7887	0

Dado que la matriz estequiométrica cuenta con una fila y tres columnas (y al haber una sola fila no cuenta con filas linealmente dependientes), el espacio nulo de esta matriz es

bidimensional. Por lo tanto, siguiendo el procedimiento planteado, se toman las dos últimas columnas de la matriz *Q* para conformar la matriz *W*, que corresponde entonces a:

Tabla 4 – Matriz W para modelo ramificado simple

W	
-0.5774	0.5774
0.7887	0.2113
0.2113	0.7887

Las columnas de *W* conforman la base ortonormal del espacio nulo de la matriz estequiométrica de este modelo. Esta base ortonormal se va utilizar para delimitar el espacio de referencia, a partir de la determinación de rangos apropiados para el sorteo de las variables aleatorias del vector *X*. Los puntos aleatorios muestreados en el espacio de referencia se van a obtener haciendo la multiplicación de un vector aleatorio *X* con distribución uniforme por la matriz *W*.

La determinación de los rangos mencionados se puede hacer resolviendo una serie de problemas de optimización, que en este trabajo se resuelven mediante el paquete de software GLPK. En estos problemas, se maximiza y minimiza cada una de las variables aleatorias del vector *X*, haciendo que los puntos muestreados cumplan con las restricciones de capacidad de flujo del modelo. Es decir, para cada variable aleatoria, se busca el máximo y mínimo valor que puede tomar si se quiere que los puntos muestreados tengan la posibilidad de estar adentro del espacio de soluciones. En este caso, se resuelven 4 problemas de optimización en total (uno de maximización y otro de minimización para cada una de las dos variables). Un problema tipo se puede encontrar en el Anexo 1. En la Tabla 5 se presentan los resultados de la resolución de estos problemas.

Tabla 5 – Extensión de la caja de muestreo para modelo ramificado simple

	min	max
x1	-2.1957	8
x2	0	12.1957

Con estos resultados, se puede proceder a calcular el volumen del espacio de referencia, que ahora está delimitado por estos rangos. El volumen (en este caso un área) se obtiene de la multiplicación de un rango por el otro, al ser perpendiculares y unitarios los vectores determinados por las columnas de *W* (base ortonormal). Por lo tanto, el volumen del espacio de referencia en este caso es de 124.34. La Ilustración 15 presenta una representación de este espacio.

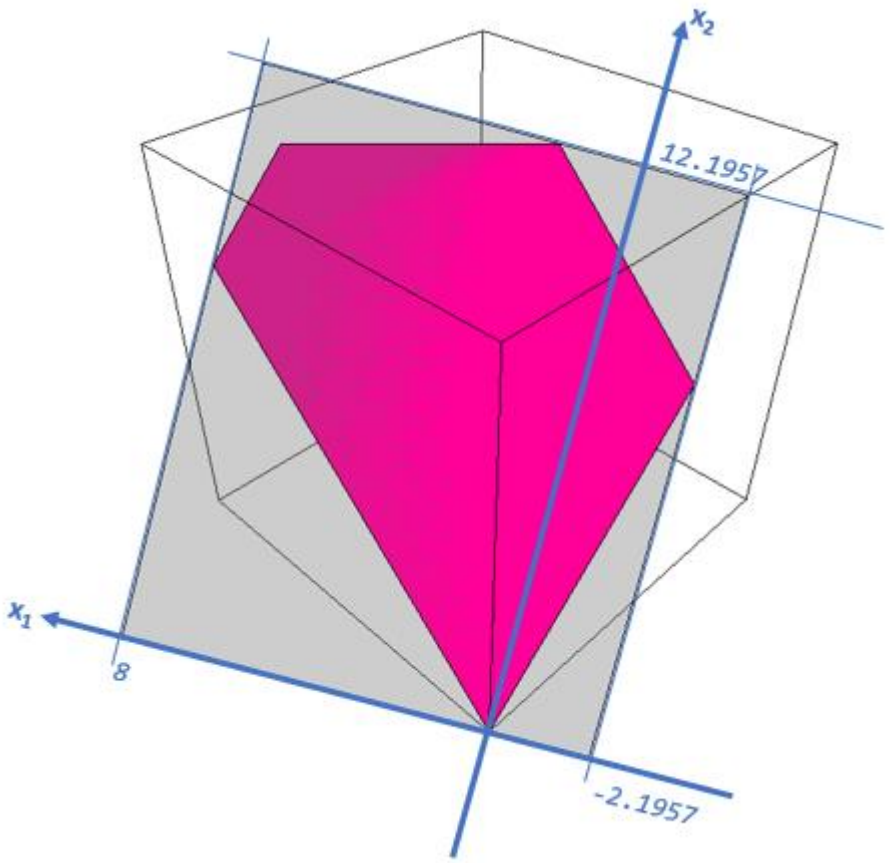


Ilustración 15 – Espacio de soluciones dentro del espacio de referencia

Implementado el algoritmo de Monte Carlo directo con 10^5 iteraciones, se obtienen los resultados que se presentan a continuación. El código se presenta en el Anexo 2.

Tabla 6 – Resultados

Cantidad de hits	55704
Hit fraction	0.55704
Volumen del espacio de muestreo	124.3436985
Volumen del espacio de soluciones	69.26441381
Desviacion estandar	0.195322106
Elapsed time (segundos)	0.821563482

Como forma de evaluar estos resultados, se calcula un intervalo de confianza del 95% para el estimador. Se elige hacer esto empleando la desigualdad de Chebyshev, que implica el cálculo de:

$$\omega_1(z,n,\beta) = \frac{z + \beta^2/2 - \beta\sqrt{\beta^2/4 + z(n-z)/n}}{n+\beta^2}$$

$$\omega_2(z,n,\beta) = \frac{z + \beta^2/2 + \beta\sqrt{\beta^2/4 + z(n-z)/n}}{n+\beta^2}$$

Donde el intervalo de confianza $(\omega_1(S,n,\delta^{-1/2}), \omega_2(S,n,\delta^{-1/2}))$ garantiza un nivel de cobertura de al menos $1-\delta$ para el estimador. Usando $\delta = 0.05$, se obtiene:

ω_1=5.500E-01

ω_2=5.641E-01

Donde ω_1 da el límite inferior para el estimador y ω_2 el límite superior. Multiplicando estos límites por el volumen de referencia, puede decirse que se espera con un 95% de confianza que el volumen real esté entre 68.3896 y 70.1364, para un ancho del intervalo de confianza de 1.7468. Dado que el volumen del espacio de soluciones se conoce, y es de exactamente $\sqrt{7500} - \sqrt{192} - \sqrt{12}$ (o aproximadamente 69.2820323 u^2), se entiende que la estimación es satisfactoria.

La Ilustración 16 presenta el resultado en forma gráfica, donde los puntos grises son los que caen afuera del espacio de soluciones, y los puntos rosados son los que caen adentro. Se puede observar cómo el espacio de referencia se ajusta al espacio de soluciones lo más posible.

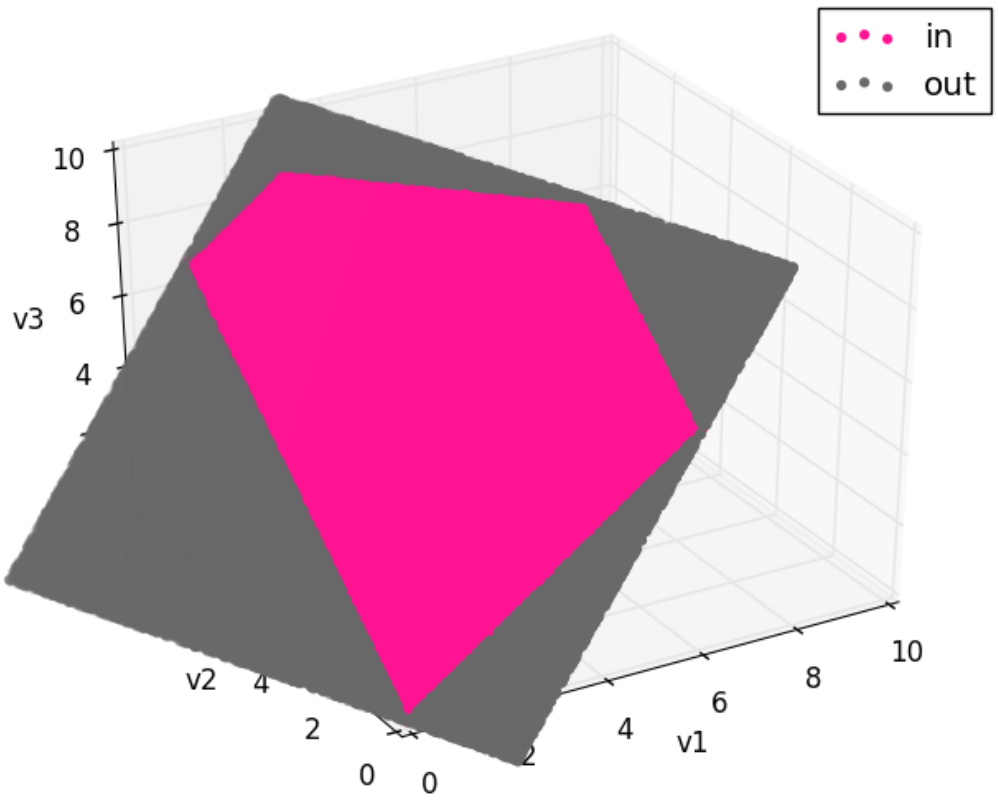


Ilustración 16 – Monte Carlo directo en modelo ramificado simple

Si se elige hacer más iteraciones, aumentando el tiempo de cálculo, la estimación mejora. En particular, para 10^7 iteraciones, se obtienen los siguientes resultados.

Tabla 7 – Resultados

Cantidad de hits	5571945
Hit fraction	0.5571945
Volumen del espacio de muestreo	124.3436985
Volumen del espacio de soluciones	69.28362491
Desviacion estandar	0.019531415
Elapsed time (segundos)	85.87890315

El intervalo de confianza del 95% en este caso va de 69.1963 a 69.3710, para un ancho de 0.1747, visiblemente mejor que en la corrida anterior.

Es interesante también destacar la importancia de optimizar el espacio de referencia para mejorar la calidad de los resultados, ajustándolo lo más posible al espacio de soluciones. Incluso en un modelo mínimo como este, la diferencia puede ser importante. Para ilustrar esto, se propone usar un espacio de referencia no ajustado al espacio de soluciones, por ejemplo uno en el que las variables aleatorias vayan de -20 a 20, para un volumen del espacio de referencia de 1600. En este caso, los resultados de hacer 10^5 iteraciones son los siguientes.

Tabla 8 – Resultados

Cantidad de hits	4218
Hit fraction	0.04218
Volumen del espacio de muestreo	1600
Volumen del espacio de soluciones	67.488
Desviacion estandar	1.016991662
Elapsed time (segundos)	1.001173973

El intervalo de confianza del 95% en este caso va de 63.0845 a 72.1845, para un ancho de 9.1000. Se puede observar cómo, al incrementar el tamaño de espacio de soluciones, la calidad de los resultados para una misma cantidad de iteraciones deja de ser satisfactoria. La Ilustración 17 presenta este último resultado en forma gráfica.

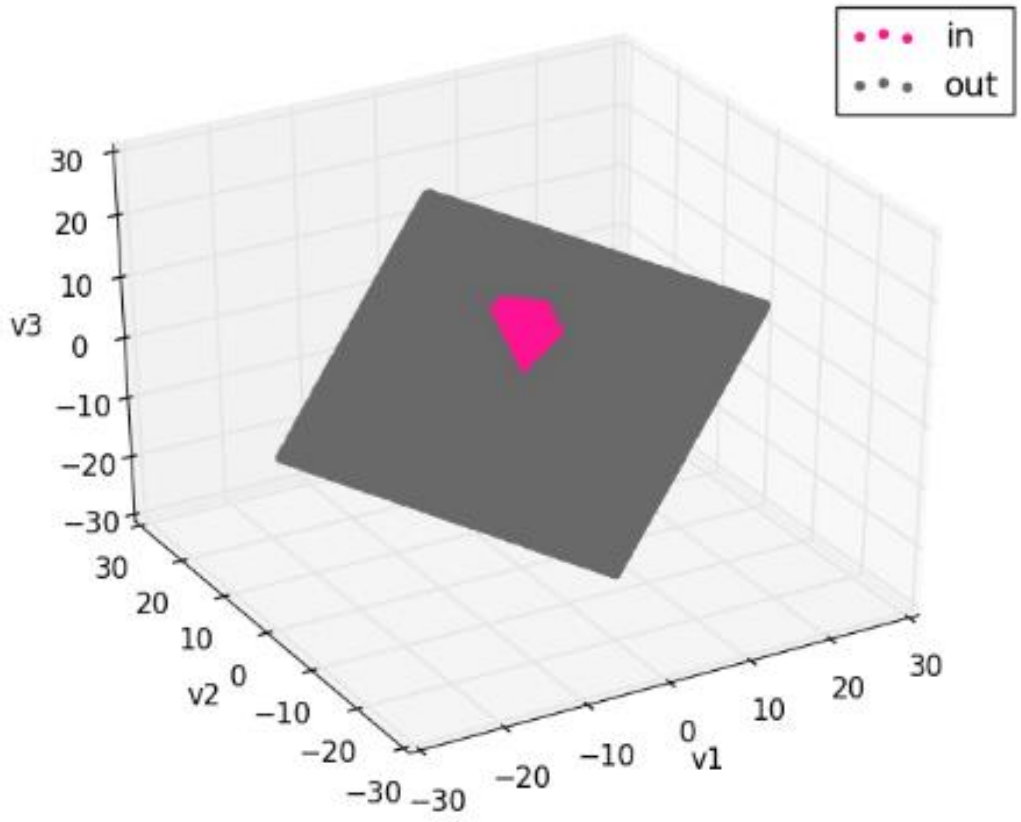


Ilustración 17 – Resultados con caja no ajustada

5.1.2. Monte Carlo directo con paralelepípedos

Teniendo en cuenta las consideraciones generales presentadas en la metodología, se destaca la importancia de generar un espacio de referencia apropiado para el espacio de soluciones del modelo. Aquí, se pretende aplicar el método de paralelepípedos presentado en la sección 4.4.1.2.2 al modelo ramificado simple, para llevar a cabo el muestreo de su espacio de soluciones mediante el método Monte Carlo directo y estimar su volumen (que, como ya se vio, corresponde a un área en este caso particular).

En primer lugar, teniendo tres restricciones de flujo máximo y dos flujos independientes (el tercero se puede determinar a partir de $V_3 = V_1 + V_2$), se tiene que el número de posibles paralelepípedos donde se puede sortear es tres. En particular, estos son:

Tabla 9 – Paralelepípedos

B_1	
0	6
8	0
8	6

B_2	
0	6
10	-6
10	0

B_3	
10	-8
0	8
10	0

- El primer paralelepípedo equivale a sortear V_2 y V_1, y definir V_3 a partir de ellos.
- El segundo paralelepípedo equivale a sortear V_3 y V_1, y definir V_2 a partir de ellos.
- El tercer paralelepípedo equivale a sortear V_3 y V_2, y definir V_1 a partir de ellos.

El flujo que sea definido a partir de los otros dos va a ser el que determine qué esquinas del paralelepípedo quedarán por fuera del espacio de soluciones. Se calcula el volumen teórico de cada paralelepípedo. La Ilustración 18 presenta las geometrías, donde se marca el espacio de soluciones en negro.

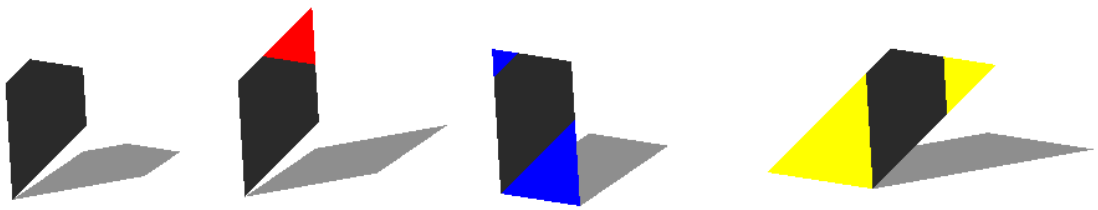


Ilustración 18 – Paralelepípedos

Los volúmenes teóricos (que en este caso son áreas) se presentan en la Tabla 10.

Tabla 10 – Volúmenes teóricos

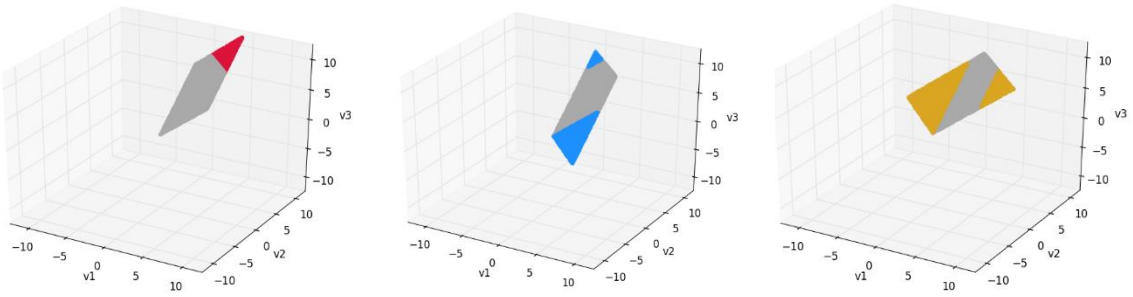
	B_1	B_2	B_3
Volumen	83.1	103.9	138.6

A continuación, se implementa en Python el código presentado en el Anexo 3 con 10^7 iteraciones, para los tres paralelepípedos. Los resultados se presentan en la Tabla 11.

Tabla 11 – Resultados

	B1	B2	B3
Semilla	123	123	123
Cantidad de iteraciones	1E+07	1E+07	1E+07
Cantidad de hits	8334863	6665057	5000822
Hit fraction	0.8335	0.6665	0.5001
Vol. paralelepípedo	83.1384	103.9230	138.5641
Vol. esp. soluciones	69.2947	69.2653	69.2934
Desv. estándar	0.0098	0.0155	0.0219
Elapsed time (segundos)	45.41	45.15	45.56

Las siguientes ilustraciones presentan los resultados de las corridas en forma gráfica, donde el espacio de soluciones se representa en gris en cada caso.



B1 B2 B3

Ilustración 19 – Resultados

Posteriormente, se calcula para cada corrida un intervalo de confianza del 95%, al igual que en la sección anterior. Los resultados son los siguientes.

Tabla 12 – Intervalos de confianza

	B1	B2	B3
Límite inferior	69.2509	69.1960	69.1954
Límite superior	69.3385	69.3346	69.3914
Ancho int. 95%	0.0876	0.1386	0.1960

Los resultados obtenidos para este modelo son satisfactorios, y se observa que el ancho del intervalo de confianza es menor cuanto menor es el tamaño del espacio de referencia, de acuerdo a lo esperado. Haciendo una comparación con el espacio de referencia generado por medio de una base ortonormal utilizado en la sección anterior, puede verse que los intervalos de confianza están en el mismo orden de magnitud. Por lo tanto, y dada la facilidad de implementación del método presentado en la sección anterior, se decide no volver a usar el método de los paralelepípedos para generar los espacios de referencia en el resto de este trabajo.

5.1.3. Uso de COBRApy sampler

5.1.3.1. Metodología

En esta sección, se pone en práctica la metodología presentada en la sección 4.4.3.2, donde se sugiere usar la herramienta de flux sampling de COBRApy para hacer una estimación del volumen de un determinado modelo metabólico. El primer paso consiste en generar un modelo metabólico alternativo cuyo espacio de soluciones coincida con la "caja" que se presenta en la sección 4.4.1.2.1. Para ello, según lo estipulado, se define un modelo alternativo con la matriz estequiométrica de la Tabla 13.

Tabla 13 – Matriz estequiométrica del modelo alternativo

	v1	**v2**	**v3**	**x1**	**x2**
A	1	1	-1	0	0
alfa	-1	0	0	-0.5774	0.5774
beta	0	-1	0	0.7887	0.2113
gama	0	0	-1	0.2113	0.7887

En azul está la matriz estequiométrica del modelo original, y en amarillo está la matriz W. Los metabolitos alfa, beta y gama son los que se definen especialmente para generar este modelo alternativo, pero no tienen ningún significado en el modelo original. Asimismo, en la Tabla 14 se detallan las restricciones de capacidad de flujo para este modelo alternativo.

Tabla 14 – Restricciones de capacidad de flujo para el modelo alternativo

	min	max
v1	-1000	1000
v2	-1000	1000
v3	-1000	1000
x1	-2.1957	8
x2	0	12.1957

Puede verse cómo las restricciones de capacidad de flujo de este modelo alternativo están en realidad restringiendo las variables aleatorias X que se multiplican por los vectores de la base ortonormal W para obtener puntos en el espacio nulo de la matriz estequiométrica original, de modo que estos puntos caigan dentro del espacio de referencia. Es decir, que la definición del modelo alternativo equivale a la definición de la caja de muestreo.

Aplicando sampling en este modelo alternativo con el método OptGpSampler, se obtienen muestras adentro de la caja (que en este caso es un rectángulo, al ser bidimensional el espacio de soluciones, según se planteó previamente). La Ilustración 20 presenta el muestreo obtenido para 10^3 muestras.

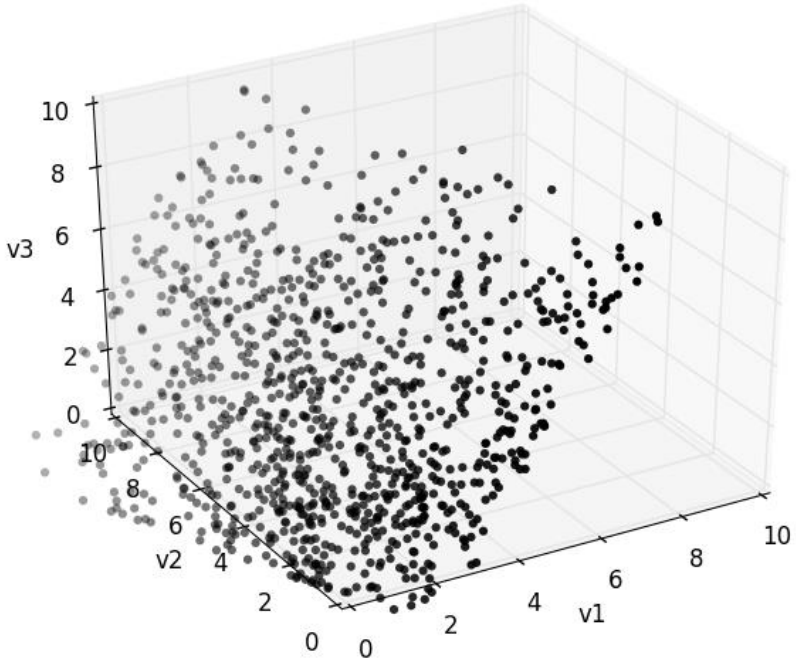


Ilustración 20 – Muestreo

Otra representación de estos datos puede hacerse en dos dimensiones, tomando como ejes x1 y x2, según se presenta a continuación.

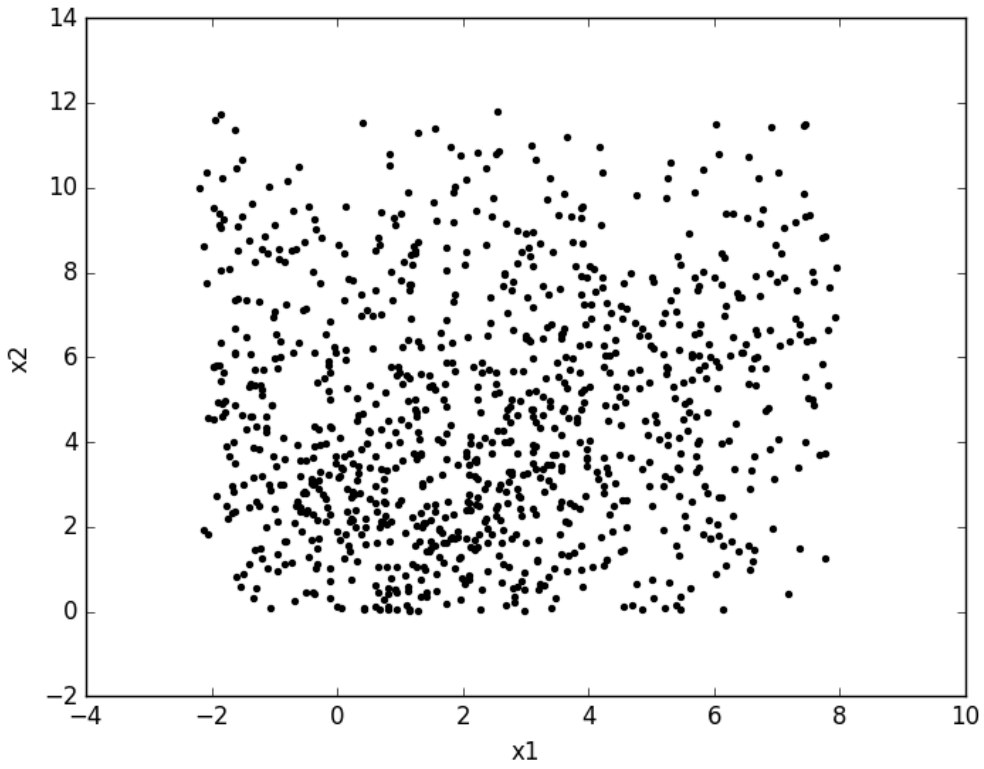


Ilustración 21 – Muestreo en dos dimensiones

El siguiente paso consiste en ver qué fracción de estas muestras cumple con la primera restricción de capacidad de flujo que se le va a añadir al modelo. En este caso, se elige como primera restricción la siguiente:

$$v_1 \leq 6$$

Se observa qué cantidad de los puntos muestreados cumplen con esta primera restricción, según se puede ver en la Ilustración 22, donde los puntos válidos están en azul y los no válidos en rojo. La cantidad de puntos que cumplen con la restricción dividida entre el total de puntos muestreados indica la fracción correspondiente a esta primera restricción.

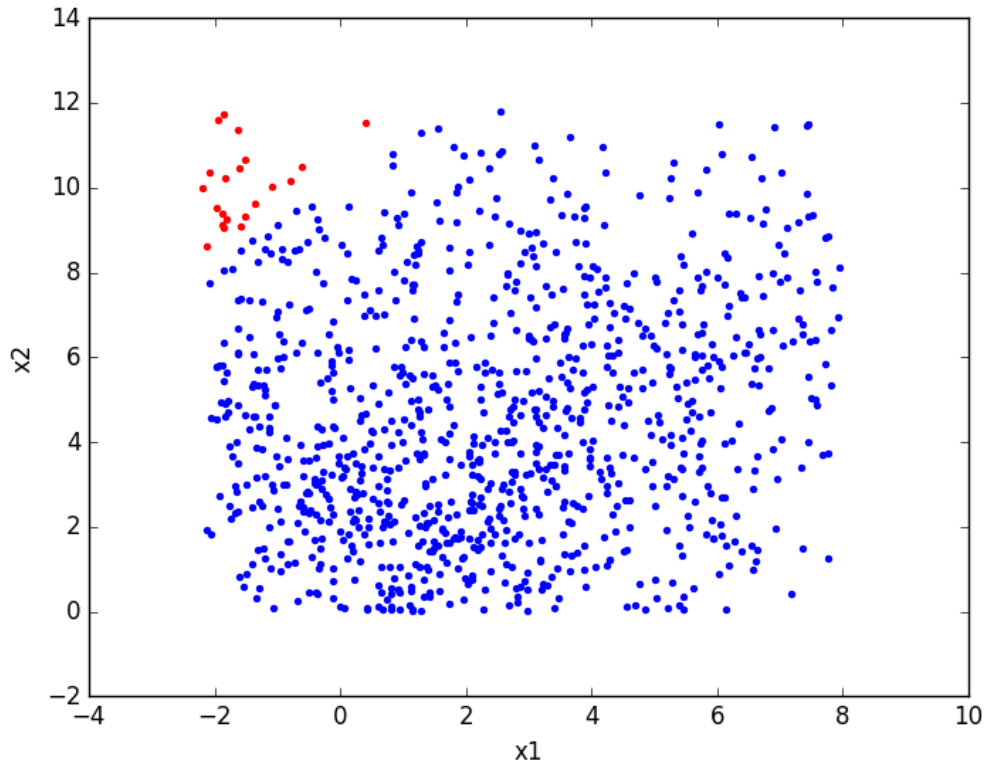


Ilustración 22 – Puntos válidos y no válidos

Posteriormente, se procede a hacer un nuevo muestreo en el mismo modelo pero ahora teniendo en cuenta también esta restricción adicional para $v1$, obteniendo nuevos puntos según se presenta en la Ilustración 23.

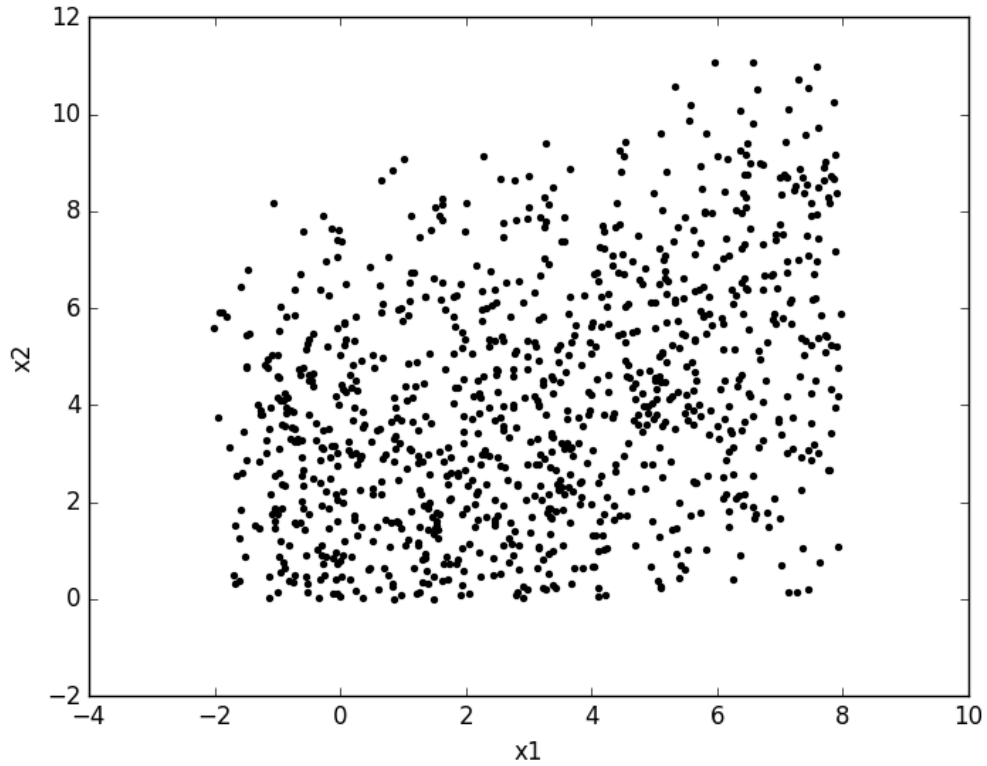


Ilustración 23 – Muestreo con restricción adicional

Seguidamente, se analiza cuántos de estos nuevos puntos cumplen con la que será la siguiente restricción, que equivale a:

$$v_1 \geq 0$$

Los puntos que cumplen con esta nueva restricción se pueden ver en la Ilustración 24, donde los puntos válidos están en azul y los no válidos en rojo.

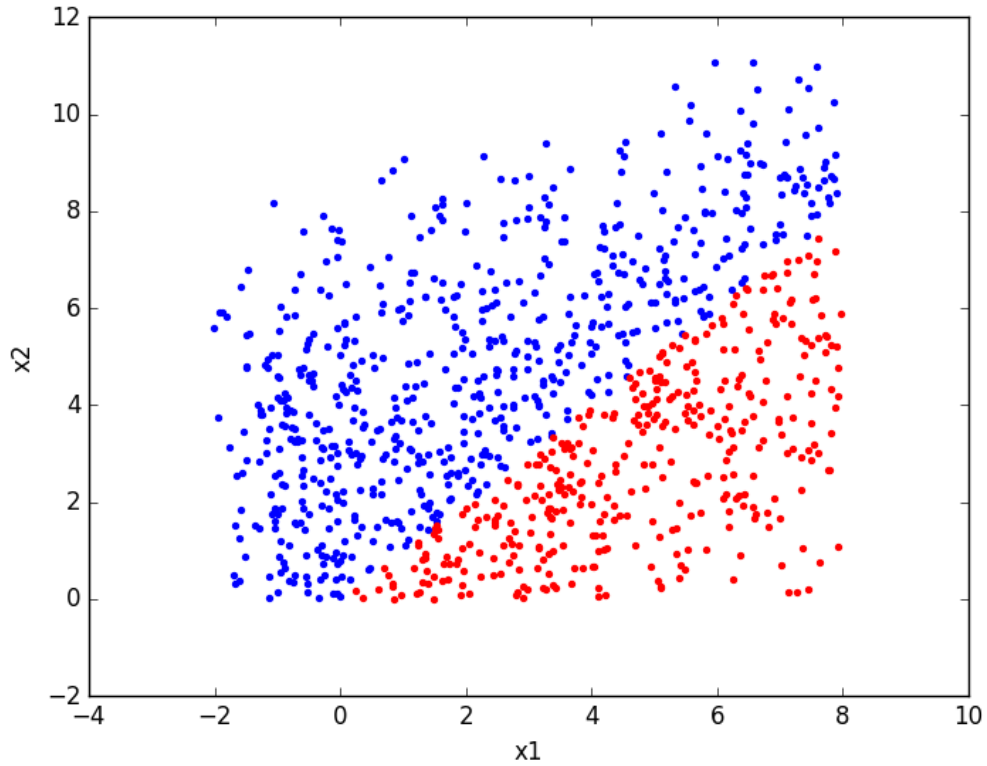


Ilustración 24 – Puntos válidos y no válidos

Se continúa con el mismo procedimiento agregando las cuatro restricciones de capacidad de flujo restantes una por una, y muestreando un nuevo set de puntos cada vez. Los resultados se presentan en el siguiente apartado.

5.1.3.2. Resultados

Se aplica el procedimiento para 10^3, 10^4 y 10^5 iteraciones en cada paso. Los siete pasos que se realizan se listan a continuación, según lo que fue presentado en la metodología:

1. Muestras en la caja de muestreo sin restricciones de flujo, se verifica cuántas cumplen con $v1 < 6$ (fracción 1).
2. Muestras en caja de muestreo con restricción de $v1 < 6$, se verifica cuántas cumplen con $v1 > 0$ (fracción 2).
3. Muestras en caja de muestreo con restricciones de $v1$, se verifica cuántas cumplen con $v2 < 8$ (fracción 3).
4. Muestras en caja de muestreo con restricciones de $v1$ y $v2 < 8$, se verifica cuántas cumplen con $v2 > 0$ (fracción 4).
5. Muestras en caja de muestreo con restricciones de $v1$ y $v2$, se verifica cuántas cumplen con $v3 < 10$ (fracción 5).
6. Muestras en caja de muestreo con restricciones de $v1$, $v2$ y $v3 < 10$, se verifica cuántas cumplen con $v3 > 0$ (fracción 6).
7. Muestras en el espacio de soluciones del modelo (caja con todas las restricciones de flujo agregadas).

El volumen de la caja de muestreo es de 124.34, según se mencionó previamente. Al multiplicar este volumen por las seis fracciones, se obtiene el volumen estimado del

espacio de soluciones (el producto de las seis fracciones indica la proporción del espacio que cumple con las seis restricciones a la vez). Los resultados obtenidos se presentan en la Tabla 15. El error se calcula en base al valor del volumen del espacio de soluciones calculado de forma analítica. De forma de revisar la calidad de las muestras, se calcula también, para cada punto muestreado, el producto de multiplicar la matriz W por el vector x de la muestra, y se comprueba que el resultado sea igual al vector v de la muestra, lo cual se cumple en todos los casos como era de esperar.

Tabla 15 – Resultados

Cantidad de muestras	Tiempo total de muestreo (segundos)	Fracción 1	Fracción 2	Fracción 3	Fracción 4	Fracción 5	Fracción 6	Volumen estimado del espacio de soluciones	% error
1000	91	0.973	0.610	0.996	0.940	0.945	1.000	65.30	5.75
10000	724	0.983	0.623	0.992	0.924	0.951	1.000	66.29	4.31
100000	6689	0.983	0.628	0.992	0.923	0.950	1.000	66.82	3.55

5.1.3.3. Consideraciones

La primera observación a considerar corresponde a los porcentajes de error encontrados para los distintos tiempos de cálculo. Es importante notar que, si bien el error disminuye al aumentar la cantidad de iteraciones, este sigue siendo mucho mayor al que se puede obtener aplicando un método Monte Carlo directo según los resultados de las secciones anteriores. La ventaja de este método, entonces, estaría en la posibilidad de aproximar un resultado para espacios de soluciones en muchas más dimensiones, donde un método Monte Carlo directo tenga demasiada dificultad para encontrar suficientes puntos válidos para hacer una estimación.

Los códigos utilizados se adjuntan en el Anexo 4. Es importante mencionar que, en el transcurso del trabajo, se encontraron varios problemas con el sampler. En particular, el sampler no tenía originalmente la capacidad de hacer muestreos en espacios de dos dimensiones, ni tampoco podía hacer muestreos en modelos sin reacciones reversibles. Estos problemas fueron corregidos por el equipo de desarrollo de COBRApy, llevando a una mejora de la herramienta. Los intercambios al respecto se presentan en el Anexo 5.

5.1.4. Muestreo por importancia

5.1.4.1. Generación del punto medio

En esta sección, se utiliza la metodología presentada en la sección 4.4.2, donde se hace uso de muestreo por importancia para calcular el volumen del espacio de soluciones del modelo ramificado simple. El primer desafío que se plantea a la hora de poner en práctica este método consiste en generar un punto medio que indique dónde se quiere tener una mayor concentración de muestras en cada dimensión del muestreo. Se elige tomar como punto medio el promedio de los puntos que surgen de maximizar y minimizar cada dimensión del muestreo. Es decir, que al hacer la optimización para encontrar los rangos correspondientes a cada dimensión, se guardan los puntos generados para máximo y mínimo para cada dimensión y se hace un promedio.

En el caso del modelo ramificado simple, se tienen los cuatro puntos que se presentan en la Tabla 16.

Tabla 16 – Puntos de maximización y minimización de cada dimensión del modelo

	Maximizando x_1:	Maximizando x_2:	Minimizando x_1:	Minimizando x_2:
x_1	8.00	1.80	-2.19	0.00
x_2	8.00	12.19	8.19	0.00

El punto medio para el uso del muestreo por importancia, que surge de promediar los cuatro puntos anteriores, se presenta en la Tabla 17.

$$x_1 = \frac{8.00 + 1.80 - 2.19 + 0.00}{4} = 1.90$$

$$x_2 = \frac{8.00 + 12.19 + 8.19 + 0.00}{4} = 7.10$$

Tabla 17 – Punto medio

x_1	1.90
x_2	7.10

Para el uso de la distribución triangular, la determinación del punto medio alcanza para aplicar muestreo por importancia. Para el caso de la distribución exponencial truncada, es necesario también determinar un parámetro adecuado para el factor λ.

5.1.4.2. Resultados con distribución triangular

Al implementar el código que se presenta en el Anexo 6 con 10^7 iteraciones, se obtienen los siguientes resultados.

Tabla 18 – Resultados

Cantidad de hits	8377425
Estimador	0.557303855
Volumen del espacio de muestreo	124.3436985
Volumen del espacio de soluciones	69.29722247
Desviacion estandar	0.022153074
Elapsed time (segundos)	2037.030046

Se observa que, si bien se obtiene una buena aproximación al volumen real del espacio de soluciones, la desviación estándar es ligeramente mayor que la obtenida en la sección 5.1.1 para 10^7 iteraciones, lo que indica que el uso de muestreo por importancia con distribución triangular puede no estar llevando a una mejora en el muestreo.

5.1.4.3. Resultados con distribución exponencial truncada

Para la distribución exponencial truncada, se utiliza el código que se presenta en el Anexo 7 con distintos valores de λ, para 10^7 iteraciones. Los resultados obtenidos se presentan en la Tabla 19.

Tabla 19 – Resultados exponencial truncada

λ	0.1	0.2	0.5	1	2
Cantidad de hits	6618578	7322082	8885209	9824259	9997257
Estimador	0.556862885	0.55692134	0.55703867	0.55695215	0.54186994
Volumen del espacio de muestreo	124.3436985	124.343698	124.343698	124.343698	124.343698
Volumen del espacio de soluciones	69.24239068	69.2496588	69.264249	69.2534901	67.3781127
Desviacion estandar	0.017056902	0.01661222	0.02615526	0.088263	1.17259103
Elapsed time (segundos)	436.9310625	426.816091	543.199017	563.98138	464.819426

Se observa cómo la desviación estándar aumenta con valores de λ muy altos, para los cuales la distribución exponencial truncada se aleja más de una distribución uniforme.

5.1.5. Cálculo de volumen integrando secciones

Como un procedimiento adicional para computar el volumen del espacio de soluciones, se propone el cálculo de distintas secciones de dicho volumen y su posterior integración. La aplicación de este procedimiento al modelo ramificado simple es importante para entender los cálculos aplicados en la sección 5.4.2.

Para tomar secciones dentro del espacio de soluciones, es necesario fijar al menos una de las variables que lo describen. Es importante recordar que el espacio de soluciones puede ser descrito tanto en términos de los flujos correspondientes al modelo, como en términos de la base ortonormal del espacio de soluciones determinada a través del método de la descomposición QR, según se vio en la sección 4.4.1.2.1. Por lo tanto, el primer paso del procedimiento corresponde a decidir si se va a fijar uno de los flujos del modelo o una de las dimensiones de la base ortonormal del espacio de soluciones. Debido a que el hecho de fijar uno de los flujos del modelo, a diferencia de fijar una de las dimensiones de la base ortonormal, tendrá un significado biológico claramente interpretable, se decide usar el procedimiento fijando uno de los flujos del modelo. Para este modelo ramificado simple, se decide tomar V_1 como el flujo a fijar a la hora de calcular las secciones.

Teniendo en cuenta que V_1 varía entre 0 y 6, se definen 13 secciones fijando V_1 en los valores: 0.0, 0.5, …, 5.5, 6.0. Para generar los modelos que corresponden a estas secciones, se aplica el procedimiento que se describe en la sección 4.3.2. En primer lugar, se determina una nueva matriz estequiométrica eliminando el flujo V_1, por lo que se pasaría de tener una matriz estequiométrica S a una matriz S^*.

	V1	V2	V3
A	-1	-1	1

	V2	V3
A	-1	1

Posteriormente, se genera el vector x_0 para cada uno de los valores en los que se fija V_1. Se recuerda que el vector x_0 corresponde a una solución cualquiera que hace que $S^* x_0 = b$, siendo b un vector del largo de la cantidad de metabolitos en el modelo que contiene ceros en todas sus entradas menos en las que corresponden a las filas del

modelo donde V_1 estaba presente. En esas entradas, el vector b contiene el valor en el que se fija V_1 multiplicado por -1 y por el coeficiente estequiométrico que tenía V_1 en esa fila.

En el caso de este modelo, para fijar V_1, el vector b siempre tendrá una sola entrada que corresponde al valor fijo de V_1 por -1. Por lo tanto, un posible x_0 es el vector que contiene 0 en la primera entrada y el valor fijo de V_1 en la segunda entrada, de modo que $S^* x_0 = b$.

Teniendo estos vectores x_0, se procede a aplicar una descomposición QR en la matriz modificada para generar una base ortonormal del espacio de soluciones de la misma, con su correspondiente caja de muestreo para poder aplicar Monte Carlo, siguiendo la metodología y teniendo en cuenta que, para aplicar las restricciones de LB y UB a los flujos que surgen de multiplicar Wx (siendo x el vector de sorteo), primero debe transformarse los mismos sumándoles el x_0.

Una vez generadas las bases y cajas de sorteo necesarias, se procede a aplicar Monte Carlo en cada uno de los 13 casos, de forma de obtener el volumen de las secciones. Es necesario notar que, al fijar un flujo en el modelo, se está bajando una dimensión, por lo que el espacio de soluciones pasa de dos dimensiones a una. Esto implica que el volumen de la caja (en este caso una longitud) va a ser exactamente igual al volumen del espacio de soluciones, por lo que en realidad no sería necesario aplicar Monte Carlo. En la Tabla 20 se dan los resultados obtenidos.

Tabla 20 – Resultados

Valor fijo V1	Volumen caja (u)	Volumen espacio sol (u)
0	11.314	11.314
0.5	11.314	11.314
1	11.314	11.314
1.5	11.314	11.314
2	11.314	11.314
2.5	10.607	10.607
3	9.8995	9.8995
3.5	9.1924	9.1924
4	8.4853	8.4853
4.5	7.7782	7.7782
5	7.0711	7.0711
5.5	6.364	6.364
6	5.6569	5.6569

Para realizar la integración y calcular el volumen del espacio de soluciones con todos los flujos libres, es importante entender primero el significado de los resultados, lo cual puede hacerse de forma gráfica (al tratarse de un modelo de bajas dimensiones).

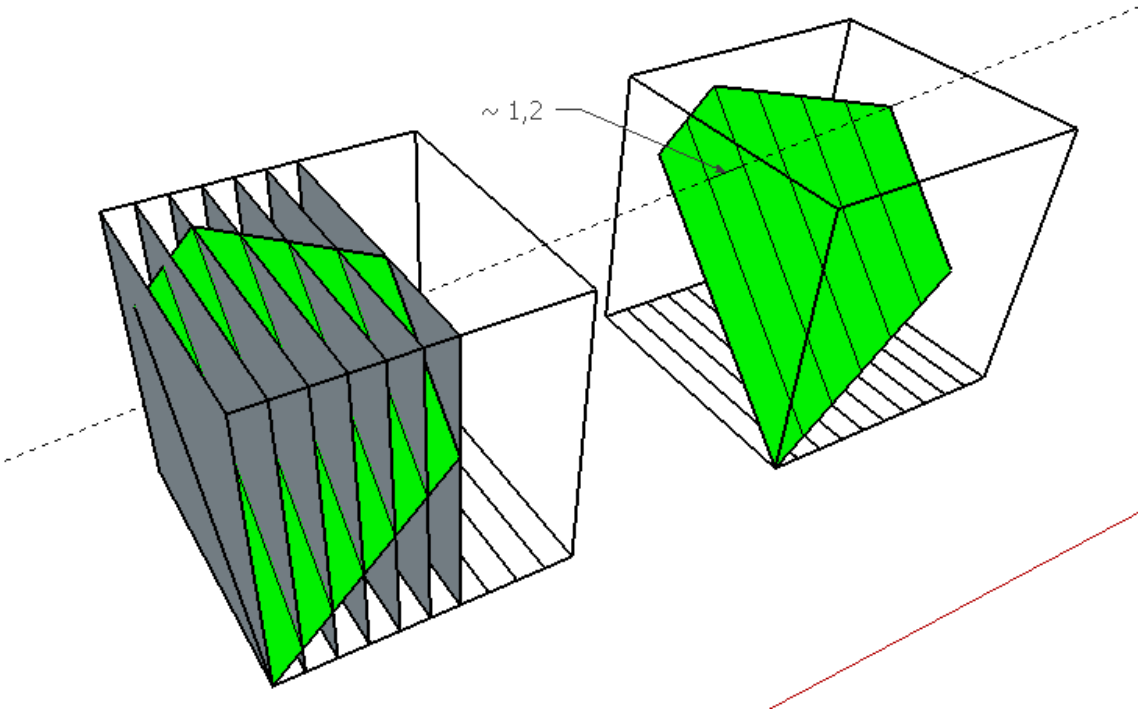


Ilustración 25 – Cortes en modelo ramificado simple

En la Ilustración 25 se puede apreciar cómo el hacer cortes sucesivos en el eje V_1 corresponde a seccionar el espacio de soluciones mediante la intersección con distintos planos de corte. Los planos de la ilustración están distribuidos a una distancia de 1 entre uno y otro. Sin embargo, al observar los cortes resultantes (que son líneas en este caso), se observa que no distan 1 entre sí, sino aproximadamente 1.2. Esto se debe a que los planos de corte son oblicuos.

Es necesario entonces poder determinar la distancia entre los cortes resultantes de manera analítica para poder hacer una correcta integración y obtener el volumen original del espacio de soluciones. Para ello, primero se deben obtener las ecuaciones que describen a los hiperplanos de corte en la base ortonormal del espacio de soluciones. De manera general, teniendo $V = Wx$ (siendo V el vector de flujos, W la matriz que convierte del espacio nulo al espacio de flujos y x el vector de sorteo en el espacio nulo), si se quiere fijar $V_1 = a$, lo que se tiene es que:

$$W_{11}x_1 + W_{12}x_2 + \ldots + W_{1n}x_n = a$$

lo cual da una ecuación para ese hiperplano. Variando a, se tienen las ecuaciones de distintos hiperplanos paralelos de corte, para los distintos valores de V_1 que se quieran fijar. Con estas ecuaciones, es sencillo calcular la distancia entre dos hiperplanos paralelos $\{x \epsilon \mathbb{R}^n | a^T x = b_1\}$ y $\{x \epsilon \mathbb{R}^n | a^T x = b_2\}$ en la base del espacio de soluciones, según:

$$dist = \frac{|b_1 - b_2|}{\|a\|_2}$$

donde $\|a\|_2$ representa la norma euclideana en $\mathbb{R}^n$: $\|a\|_2 = \sqrt{a_1^2 + \cdots + a_n^2}$.

Teniendo en cuenta estas consideraciones, se tiene que la distancia entre los hiperplanos correspondientes a fijar V_1 en valores de 0, 1, …, 6 es, en efecto, de 1.22. Para valores fijos en 0, 0.5, …, 6, la distancia es de 0.61. Integrando los cortes teniendo en cuenta esta distancia, se tiene una aproximación al valor del volumen del espacio de soluciones de 69.01 u^2, el cual es consistente con el valor de 69.28 u^2 calculado analíticamente en la sección 4.1.1.2.

5.2. Construcción y validación del modelo núcleo compacto de E.coli (E.coli compact core)

5.2.1. Presentación del modelo

Se construye el modelo núcleo compacto de E.coli a partir del modelo núcleo de E.coli, intentando reducir el número de reacciones e intermediarios sin alterar el funcionamiento. La Ilustración 26 presenta los flujos del modelo resultante en forma gráfica. En el Anexo 8 se documenta la obtención de cada reacción del núcleo compacto de E.coli, compactando las reacciones del modelo núcleo de E.coli.

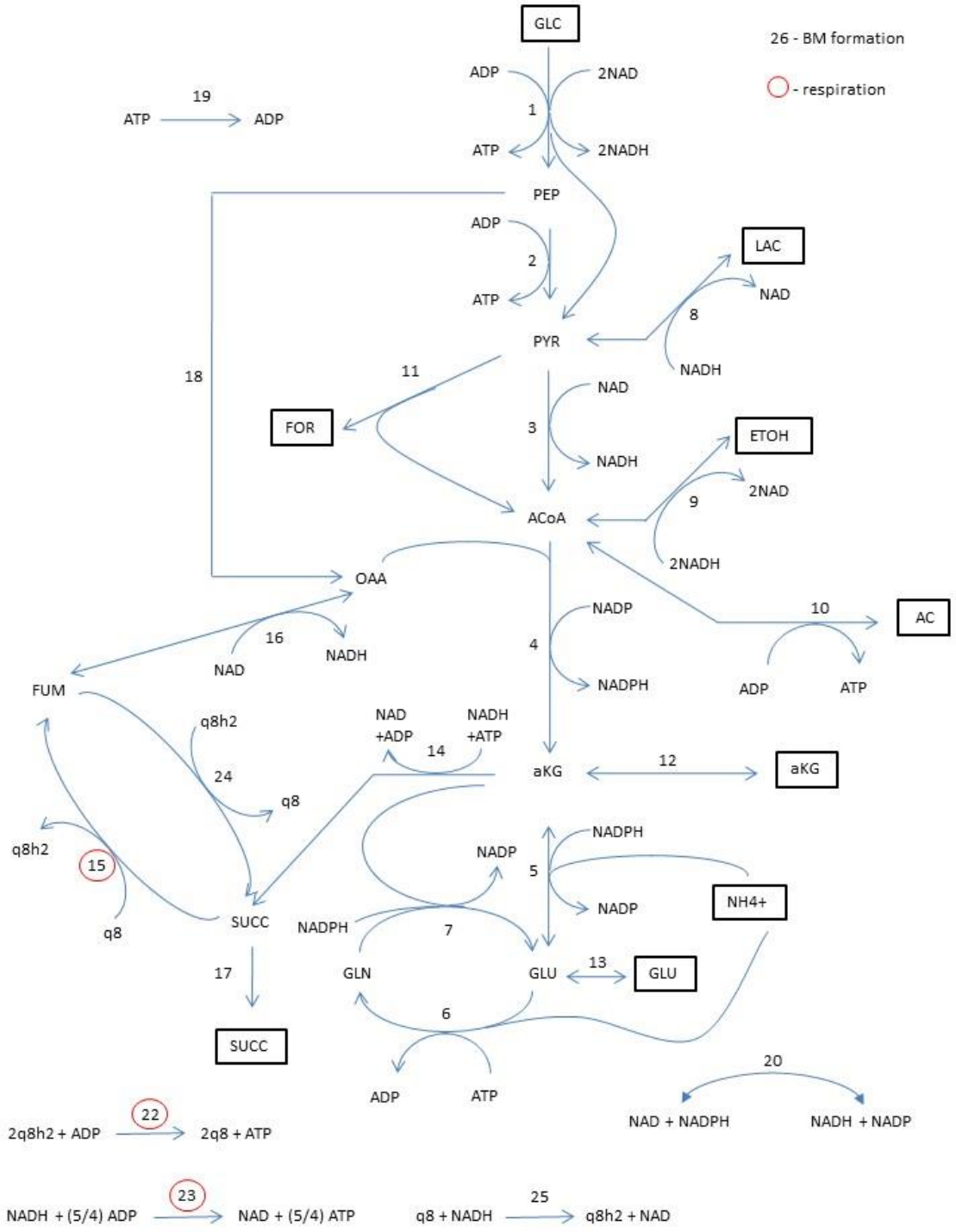


Ilustración 26 – Modelo núcleo compacto de E.coli

Las 9 especies químicas que están encerradas en un rectángulo son metabolitos externos. Sus concentraciones se asumen fijas, por lo que son consideradas como parámetros. El sistema cuenta con un total de 17 metabolitos internos y 25 flujos. Las reacciones 20 y 21, que aparecen en el Anexo 8, se fusionaron en una que en adelante llamaremos reacción 20 (la reacción 21 no existirá en la numeración del modelo). Las fermentaciones presentes en el modelo se identifican como: $V08$, $V09$, $V10$, $V11$, $V12$, $V13$ y $V17$.

El núcleo compacto de E.coli tiene tres reacciones reversibles: 5, 16 y 20. Además de las tres reacciones mencionadas, 7 de las 9 fermentaciones también son reversibles. Pero, como se usará glucosa como única fuente de carbono, las concentraciones de los productos de fermentación serán cero y todas las fermentaciones trabajarán en condiciones de irreversibilidad. Esta información se incorpora en el modelo mediante los valores de las cotas superiores e inferiores de los flujos de las reacciones de fermentación.

Asimismo, se acota superiormente la entrada de glucosa a una cota superior (upper bound, UB) de 10. A la reacción $V19$ se le asigna una cota inferior (lower bound, LB) de 8.39 como requerimiento de ATP de mantenimiento, al igual que en el modelo núcleo de E.coli. La Tabla 21 presenta los LB y UB de los flujos, donde un flujo de -1000 o 1000 implica que la reacción no está restringida inferior o superiormente, respectivamente. Esto se demostrará mediante FVA en las secciones 5.2.3 y 5.2.4.

Tabla 21 – LB y UB en núcleo compacto de E.coli

	LB	UB
V01	0	10
V02	0	1000
V03	0	1000
V04	0	1000
V05	-1000	1000
V06	0	1000
V07	0	1000
V08	0	1000
V09	0	1000
V10	0	1000
V11	0	1000
V12	0	1000
V13	0	1000
V14	0	1000
V15	0	1000
V16	-1000	1000
V17	0	1000
V18	0	1000
V19	8.39	1000
V20	-1000	1000
V22	0	1000
V23	0	1000
V24	0	1000
V25	0	1000
VBM	0	1000

La matriz estequiométrica del modelo se presenta en la Tabla 22.

Tabla 22 – Matriz estequiométrica del modelo núcleo compacto de E.coli

	V01	V02	V03	V04	V05	V06	V07	V08	V09	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V22	V23	V24	V25	VBM
ACoA	0	0	1	-1	0	0	0	0	-1	-1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	-3.748
ADP	-1	-1	0	0	0	1	0	0	0	-1	0	0	0	-1	0	0	0	0	1	0	-1	-1.25	0	0	59.81
aKG	0	0	0	1	-1	0	-1	0	0	0	0	-1	0	-1	0	0	0	0	0	0	0	0	0	0	4.1182
ATP	1	1	0	0	0	-1	0	0	0	1	0	0	0	1	0	0	0	0	-1	0	1	1.25	0	0	-59.81
FUM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-1	0	0	0	0	0	0	-1	0	0
GLN	0	0	0	0	0	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.256
GLU	0	0	0	0	1	-1	2	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	-4.941
NAD	-2	0	-1	0	0	0	0	1	2	0	0	0	0	-1	0	-1	0	0	0	-1	0	1	0	1	-3.547
NADH	2	0	1	0	0	0	0	-1	-2	0	0	0	0	1	0	1	0	0	0	1	0	-1	0	-1	3.547
NADP	0	0	0	-1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	13.028
NADPH	0	0	0	1	-1	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	-13.03
OAA	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	-1.787
PEP	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	-0.519
PYR	1	1	-1	0	0	0	0	-1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	-2.833
q8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	2	0	1	-1	0
q8h2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	-2	0	-1	1	0
SUCC	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-1	0	-1	0	0	0	0	0	1	0	0

Para simular condiciones anaeróbicas en el modelo, es necesario llevar a cero el UB de las reacciones $V15$, $V22$ y $V23$.

5.2.2. Eliminación de filas linealmente dependientes de la matriz estequiométrica

Se aplica la metodología presentada en la sección 4.3.1 al modelo núcleo compacto de E.coli. Este modelo cuenta con una matriz estequiométrica que comprende un total de 25 flujos y 17 metabolitos. El rango de la matriz estequiométrica se calcula en Python:

```
rango=numpy.linalg.matrix_rank(S)
```

Lo que devuelve un valor de 13, indicando que hay 4 filas que se pueden expresar como combinaciones lineales de las otras 13 filas. Siguiendo el procedimiento:

```
St=numpy.matrix.transpose(S)

Q,R,P=scipy.linalg.qr(St,mode="full",pivoting=True)
```

La matriz P indica que las 4 filas son las correspondientes a los metabolitos: ATP, NADH, NADP y q8h2. Las filas que corresponden a estos metabolitos pueden eliminarse de la matriz sin cambiar las restricciones que implica exigir SV=0. Eliminando dichos metabolitos del modelo, se pasa a trabajar con una matriz de rango completo, que tiene un total de 25 flujos y 13 metabolitos, con una dimensión de su espacio de soluciones de 12.

Adicionalmente, se puede observar, como se describe en la sección 4.3.1, que la existencia de filas linealmente dependientes es causada por relaciones de conservación entre determinados metabolitos. En el modelo núcleo compacto de E.coli, estas relaciones de conservación serían:

- $ATP + ADP = At$
- $NADH + NAD = NADt$
- $NADPH + NADP = NADPt$
- $q8h2 + q8 = qt$

Donde At, $NADt$, $NADPt$ y qt son las concentraciones totales de las especies que se conservan.

5.2.3. Cálculo de valores máximos y mínimos que las reacciones pueden alcanzar

El primer paso consiste en hacer FVA en el modelo núcleo compacto de E.coli para verificar que todos los flujos se prenden bajo alguna circunstancia. Este análisis se hace sobre el modelo en sus condiciones originales, es decir, con todas las vías disponibles (condición aeróbica) y con un flujo de entrada de glucosa entre 0 y 10. Los resultados del análisis FVA se presentan en la Ilustración 27, donde se pueden observar los rangos en los que se ubica cada uno de los flujos.

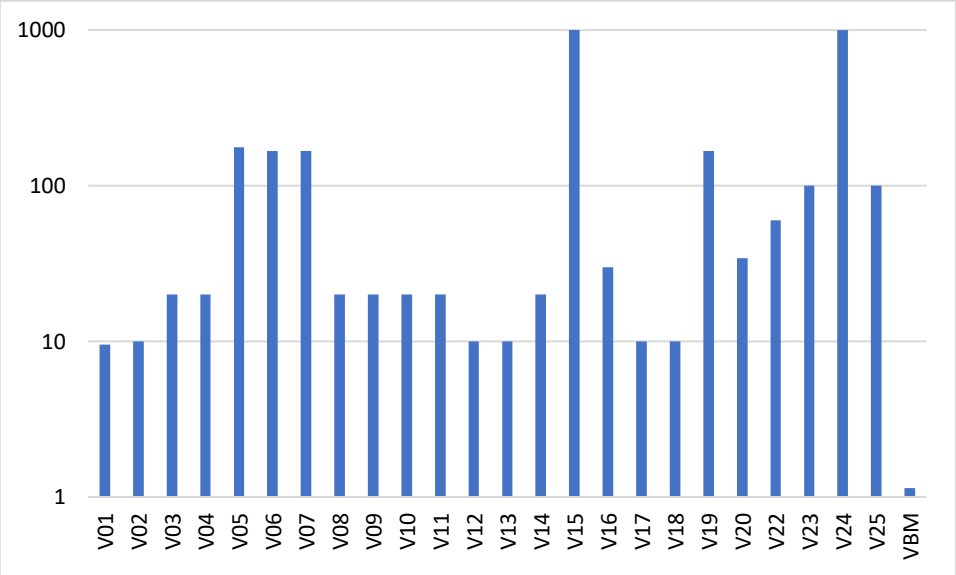


Ilustración 27 – Rangos de funcionamiento de los flujos

Se observa que los rangos son mayores a cero para todos los flujos, lo que indica que cada una de las reacciones tiene la capacidad de prenderse bajo alguna circunstancia.

Este análisis se llevó a cabo mediante la herramienta "flux_variability_analysis" de COBRApy (descripta en la sección 4.2), la cual encuentra el rango (valor máximo y valor mínimo) para cada flujo en el modelo. Se puede imponer una restricción adicional como maximizar un flujo en particular, por ejemplo el correspondiente a la reacción de crecimiento, para ver el rango que puede tomar cada flujo en el modelo cumpliendo esta condición. En este caso no se asignó ninguna restricción adicional. El análisis FVA fue descrito previamente en la sección 2.2.2.

Adicionalmente, de manera de comprobar el funcionamiento de la herramienta "flux_variability_analysis" de COBRApy, se decidió llevar a cabo el mismo análisis usando el software GLPK, para ver si se llegaba a los mismos resultados. Para ello, se planteó en GLPK un problema según se explicita en la sección 4.2. Los resultados usando GLPK fueron los mismos que usando la herramienta de COBRApy, como era esperado.

5.2.4. Detección y eliminación de ciclos fútiles

El siguiente paso en el análisis del modelo consiste en eliminar los ciclos fútiles presentes en el mismo. Se denomina ciclos fútiles a aquellas vías metabólicas que pueden estar activas cuando todas las reacciones externas se encuentran apagadas. No

tienen ningún efecto más que la disipación de energía. En particular, esto puede ocurrir cuando un sustrato es convertido en producto por una vía y luego se resintetiza el sustrato por la otra vía. Para la detección de estos ciclos en el modelo, se cierran todas las reacciones externas y se hace un FVA, analizando si existen flujos que puedan ser distintos de cero en las condiciones planteadas.

En el modelo núcleo compacto de E.coli, las reacciones externas corresponden a: $V01$, $V05$, $V06$, $V08$, $V09$, $V10$, $V11$, $V12$, $V13$ y $V17$. Llevando a cero el LB y el UB para estas reacciones, se resuelven nuevamente los problemas de programación lineal planteados en el apartado anterior, usando la herramienta "flux_variability_analysis" de COBRApy. El LB del módulo $V19$ se bajó de 8.39 a cero para que el planteo fuera factible. Se obtienen los resultados para los rangos de los flujos que se presentan en la Ilustración 28.

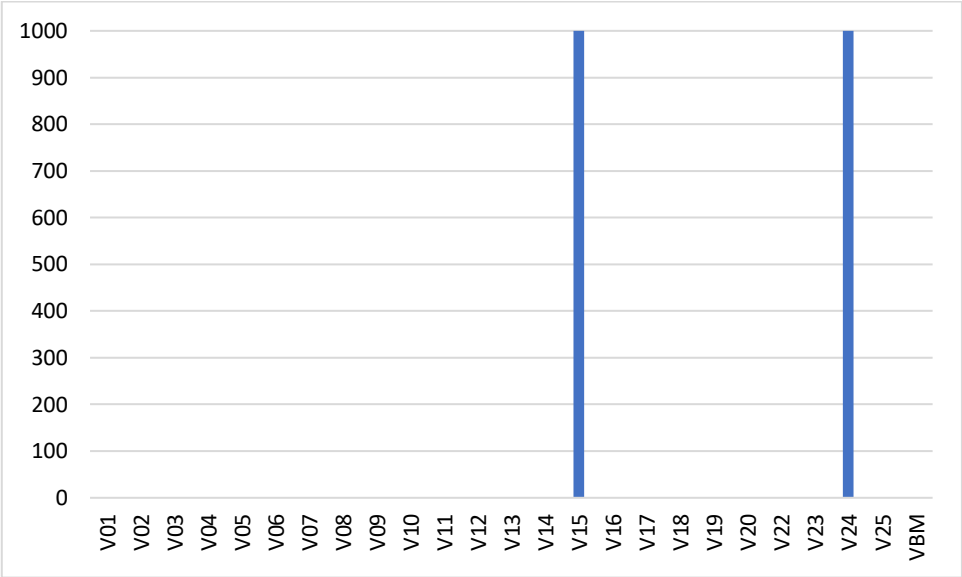


Ilustración 28 – Rangos sin reacciones externas

Se observa la existencia de un ciclo fútil, que corresponde a los módulos $V15$ y $V24$. Estas vías pueden estar activas aunque no haya entradas ni salidas del sistema. El módulo $V15$ corresponde a la oxidación de succinato a fumarato, mientras que el módulo $V24$ corresponde a la reducción de fumarato a succinato. Se decide eliminar este ciclo fútil restringiendo el UB de $V15$ en el modelo.

Para determinar el valor a asignar al UB de $V15$, se elige maximizar todas las reacciones del sistema sin restringir $V15$, y luego restringir $V15$ en el menor valor compatible con que todas las reacciones puedan adoptar el valor máximo calculado. El procedimiento es el siguiente:

1. Maximizar todas las reacciones del sistema, menos las dos del ciclo fútil, sin restringir $V15$ (23 maximizaciones)
2. Minimizar $V15$ sujeto a que cada una de las reacciones del sistema (por separado) esté acotada inferiormente por el máximo determinado en 1 (23 minimizaciones)
3. Elegir el mayor valor de los 23 mínimos de $V15$ calculados en 2
4. Emplear el valor elegido en 3 como cota superior de $V15$

Siguiendo el procedimiento y procesando los modelos en GLPK, se llega a un valor de UB de $V15$ de 20. Se tiene en cuenta que, con un valor de 7.41 como UB de $V15$, VBM puede llegar al máximo, pero un valor de 20 logra que todas las reacciones puedan llegar a su máximo correspondiente.

Si se restringe el UB de $V15$ a 20, la cota superior de 1000 para todas las otras reacciones (menos $V01$) implica que las reacciones no están restringidas superiormente, tal como se mencionó en la sección 5.2.1.

5.2.5. Comparación con núcleo de E.coli empleando FBA y FVA

5.2.5.1. Introducción

A efectos de verificar el funcionamiento del modelo núcleo compacto de E.coli, se usa FBA y FVA para hacer una comparación entre el mismo y el modelo núcleo original. En la sección 5.2.5.2 se presentan los resultados de los estudios llevados a cabo en ambos modelos, que comprenden lo siguiente:

- FBA maximizando crecimiento en condiciones aeróbicas y anaeróbicas, recordando que las condiciones anaeróbicas implican hacer cero el UB de las reacciones $V15$, $V22$ y $V23$. Analizar los patrones de flujo mediante FVA.
- FBA maximizando la formación de los productos de fermentación: LAC, EtOH, AC, FOR, aKG, GLU y SUCC (maximizando uno a la vez). Analizar los patrones de flujo mediante FVA.
- FBA maximizando consumo de oxígeno en condiciones aeróbicas y FBA maximizando consumo de ATP en condiciones aeróbicas y anaeróbicas.

Los modelos E.coli core y E.coli compact core se presentan en en el formato correspondiente a GLPK y a COBRApy en Anexo 9 y Anexo 10.

5.2.5.2. Resultados

5.2.5.2.1. Máximo crecimiento en condiciones aeróbicas y anaeróbicas

Se procede a hacer un FBA en el modelo núcleo compacto de E.coli para obtener el máximo crecimiento al que se puede llegar, tanto en condiciones aeróbicas como anaeróbicas. En condiciones aeróbicas, los módulos $V15$, $V22$ y $V23$ tienen un LB de cero (son irreversibles) y un UB de 1000 (no están acotados). En condiciones anaeróbicas, por otro lado, los módulos $V15$, $V22$ y $V23$ tienen un LB=UB=0, es decir, que se inactivan estas reacciones, que son las asociadas al proceso de respiración celular.

El FBA se lleva a cabo mediante la herramienta “model.optimize()” de COBRApy (ver sección 4.2), habiendo asignado previamente a la reacción de crecimiento (VBM) como flujo a optimizar en el modelo. FBA fue descrito previamente en la sección 2.2.1.

Resolviendo este problema en condiciones aeróbicas, se obtiene un crecimiento máximo de 1.14. Para condiciones anaeróbicas, el crecimiento máximo es menor, siendo de 0.32. Los mismos resultados se obtienen al resolver el problema mediante GLPK, validando así la herramienta de COBRApy.

En el modelo núcleo original, se tiene un valor de 0.87 para el máximo crecimiento en condiciones aeróbicas con un flujo máximo de entrada de glucosa de 10. Para el mismo flujo de entrada de glucosa en condiciones anaeróbicas, el crecimiento máximo es de 0.21. Estos valores se obtienen de [5]. Se observa que los valores para el modelo núcleo original no son tan distintos de los obtenidos para el núcleo compacto de E.coli, teniendo en cuenta la gran simplificación que se le hizo al modelo original para llevarlo a su versión compacta. En particular, se observa que el crecimiento máximo en condiciones anaeróbicas para el modelo núcleo compacto de E.coli corresponde al 28% del crecimiento máximo en condiciones aeróbicas para el mismo modelo, mientras que esta relación es del 24% para el modelo núcleo, lo que indica un comportamiento muy similar en ambos modelos en este sentido.

De manera de comprobar los valores obtenidos de la literatura, se aplica FBA al modelo núcleo mediante la herramienta "model.optimize()" de COBRApy. El modelo se presenta en formato COBRApy en el Anexo 9, pero también puede cargarse directamente a través del comando:

```
import cobra.test

model=cobra.test.create_test_model("textbook")
```

Para simular un flujo máximo de entrada de glucosa de 10, se hace -10 el LB de la reacción "EX_glc(e)" (la reacción simula una salida, por lo que la entrada al sistema tiene un valor negativo). Para simular las condiciones anaeróbicas, se lleva a cero el valor de la reacción "EX_o2(e)", la cual corresponde al intercambio de oxígeno del sistema con el medio. En condiciones aeróbicas, esta reacción queda libre para tomar valores de -1000 a 1000. Siguiendo estos criterios y aplicando FBA, los valores de crecimiento máximo obtenidos son los mismos que los presentados en [5], comprobando así la validez de los valores de crecimiento publicados.

Posteriormente, se analizan también los patrones de flujo en el modelo núcleo compacto de E.coli bajo la condición de máximo crecimiento (tanto en condiciones aeróbicas como anaeróbicas). Usando la herramienta "flux_variability_analysis" de COBRApy para resolver estos problemas de FVA, se obtienen los resultados que se presentan en la Ilustración 29 y en la Ilustración 30. En estas ilustraciones, se puede observar cómo el valor de VBM máximo es menor en condiciones anaeróbicas que en condiciones aeróbicas, de acuerdo con lo presentado. Asimismo, se observa cómo los flujos de las fermentaciones están cerrados en condiciones aeróbicas para maximizar la producción de biomasa, pero no así en condiciones anaeróbicas.

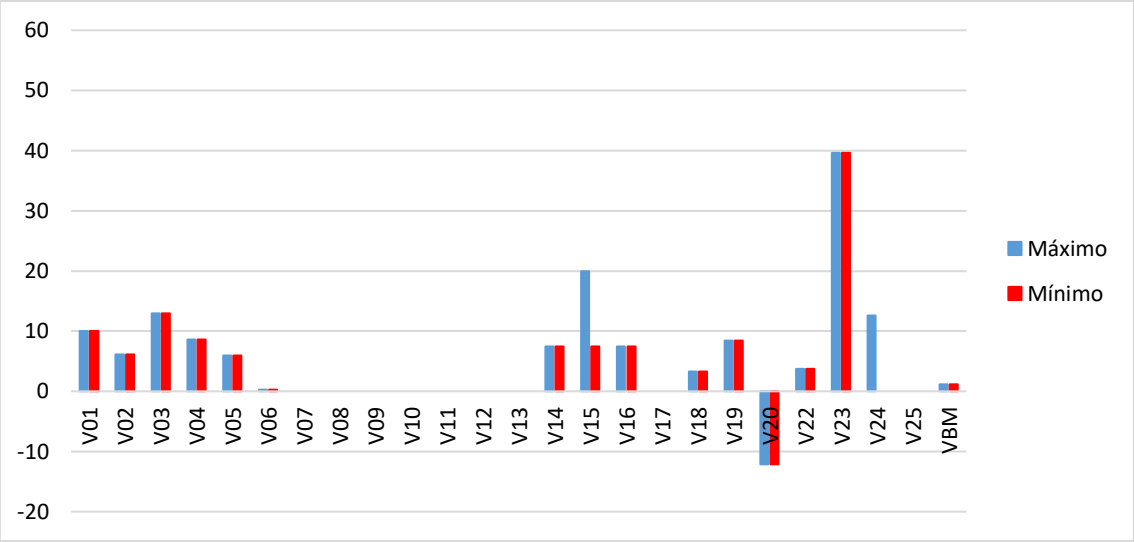


Ilustración 29 – Condiciones aeróbicas

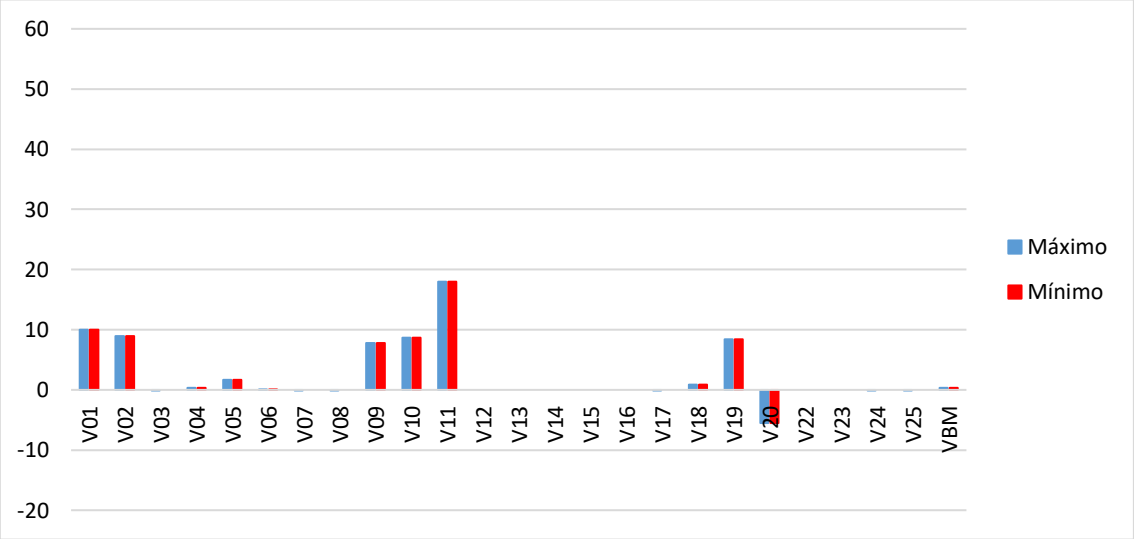


Ilustración 30 – Condiciones anaeróbicas

5.2.5.2.2. Maximización de formación de productos de fermentación

Se formula una maximización en el modelo núcleo compacto de E.coli de la velocidad de formación de los productos externos del metabolismo, haciendo FBA de la misma forma que en la sección anterior, pero eligiendo como función objetivo los flujos correspondientes a las fermentaciones y no el flujo de crecimiento. Este análisis se lleva a cabo tanto para el caso aeróbico como para el anaeróbico. Se empleó también el mismo análisis en el modelo núcleo original, de forma de poder comparar ambos modelos. La Tabla 23 y la Tabla 24 presentan los resultados obtenidos.

Tabla 23 – Condiciones aeróbicas

Producto de fermentación	Flujo en compact core	Flujo en núcleo original	Valor de flujo máximo en compact core	Valor de flujo máximo en núcleo original
LAC	V08	EX_lac_D(e)	20	20
EtOH	V09	EX_etoh(e)	20	20
AC	V10	EX_ac(e)	20	20
FOR	V11	EX_for(e)	20	40
aKG	V12	EX_akg(e)	10	10
GLU	V13	EX_glu_L(e)	10	10
SUCC	V17	EX_succ(e)	10	16.4

Tabla 24 – Condiciones anaeróbicas

Producto de fermentación	Flujo en compact core	Flujo en núcleo original	Valor de flujo máximo en compact core	Valor de flujo máximo en núcleo original
LAC	V08	EX_lac_D(e)	20	20
EtOH	V09	EX_etoh(e)	20	20
AC	V10	EX_ac(e)	10	10
FOR	V11	EX_for(e)	20	20
aKG	V12	EX_akg(e)	4	4
GLU	V13	EX_glu_L(e)	5	5
SUCC	V17	EX_succ(e)	10	13.9

En el caso aeróbico, se observa que los resultados son equivalentes en los dos modelos para LAC, EtOH, AC, aKG y GLU. Para FOR y SUCC, los resultados difieren. En el caso anaeróbico, los resultados solo difieren para SUCC.

Aplicando FVA mediante la herramienta “flux_variability_analysis” de COBRApy, se pueden observar los patrones de flujo para las distintas maximizaciones de productos. La Tabla 25 y la Tabla 26 presentan estos resultados en el caso aeróbico y anaeróbico, donde los flujos que se maximizan se resaltan en azul.

Tabla 25 – Condiciones aeróbicas

	LAC		ETOH		AC		FOR		aKG		GLU		SUCC	
	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN
V01	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
V02	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	0.0	0.0	0.0	0.0	0.0	0.0
V03	0.0	0.0	20.0	20.0	20.0	0.0	0.0	0.0	10.0	0.0	10.0	0.0	10.0	0.0
V04	0.0	0.0	0.0	0.0	0.0	0.0	20.0	0.0	10.0	10.0	10.0	10.0	10.0	0.0
V05	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10.0	0.0	0.0	0.0
V06	11.6	0.0	11.6	0.0	81.6	0.0	141.6	0.0	51.6	0.0	39.1	0.0	74.1	0.0
V07	11.6	0.0	11.6	0.0	81.6	0.0	141.6	0.0	51.6	0.0	39.1	0.0	74.1	0.0
V08	**20.0**	**20.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	8.3	0.0
V09	0.0	0.0	**20.0**	**20.0**	0.0	0.0	16.0	0.0	0.0	0.0	0.0	0.0	8.3	0.0
V10	0.0	0.0	0.0	0.0	**20.0**	**20.0**	20.0	0.0	0.0	0.0	0.0	0.0	10.0	0.0
V11	0.0	0.0	0.0	0.0	20.0	0.0	**20.0**	**20.0**	10.0	0.0	10.0	0.0	10.0	0.0
V12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**10.0**	**10.0**	0.0	0.0	0.0	0.0
V13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**10.0**	**10.0**	0.0	0.0
V14	0.0	0.0	0.0	0.0	0.0	0.0	20.0	0.0	0.0	0.0	0.0	0.0	10.0	0.0
V15	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0
V16	0.0	0.0	0.0	0.0	0.0	0.0	20.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
V17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**10.0**	**10.0**
V18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10.0	10.0	10.0	10.0	10.0	10.0
V19	20.0	8.4	20.0	8.4	90.0	8.4	150.0	8.4	60.0	8.4	47.5	8.4	82.5	8.4
V20	0.0	0.0	0.0	0.0	0.0	0.0	20.0	0.0	10.0	10.0	0.0	0.0	10.0	0.0
V22	0.0	0.0	0.0	0.0	20.0	0.0	50.0	0.0	20.0	0.0	15.0	0.0	25.0	0.0
V23	0.0	0.0	0.0	0.0	40.0	0.0	80.0	0.0	40.0	0.0	30.0	0.0	50.0	0.0
V24	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	20.0	0.0	30.0	0.0
V25	0.0	0.0	0.0	0.0	40.0	0.0	80.0	0.0	40.0	0.0	30.0	0.0	50.0	0.0
VBM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Tabla 26 – Condiciones anaeróbicas

	LAC		ETOH		AC		FOR		aKG		GLU		SUCC	
	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN
V01	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
V02	10.0	10.0	10.0	10.0	10.0	0.0	10.0	10.0	6.0	0.0	5.0	0.0	0.0	0.0
V03	0.0	0.0	20.0	20.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10.0	0.0
V04	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	4.0	5.0	5.0	3.3	0.0
V05	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0
V06	11.6	0.0	11.6	0.0	21.6	0.0	21.6	0.0	7.6	0.0	6.6	0.0	11.6	0.0
V07	11.6	0.0	11.6	0.0	21.6	0.0	21.6	0.0	7.6	0.0	6.6	0.0	11.6	0.0
V08	**20.0**	**20.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	8.3	0.0
V09	0.0	0.0	**20.0**	**20.0**	10.0	0.0	10.0	10.0	12.0	6.0	10.0	5.0	8.3	0.0
V10	0.0	0.0	0.0	0.0	**10.0**	**10.0**	10.0	10.0	0.0	0.0	0.0	0.0	10.0	0.0
V11	0.0	0.0	0.0	0.0	20.0	10.0	**20.0**	**20.0**	16.0	10.0	15.0	10.0	10.0	0.0
V12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**4.0**	**4.0**	0.0	0.0	0.0	0.0
V13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**5.0**	**5.0**	0.0	0.0
V14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.3	0.0
V15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
V16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
V17	0.0	0.0	0.0	0.0	10.0	0.0	0.0	0.0	6.0	0.0	5.0	0.0	**10.0**	**10.0**
V18	0.0	0.0	0.0	0.0	10.0	0.0	0.0	0.0	10.0	4.0	10.0	5.0	10.0	10.0
V19	20.0	8.4	20.0	8.4	30.0	8.4	30.0	8.4	16.0	8.4	15.0	8.4	20.0	8.4
V20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	4.0	0.0	0.0	3.3	0.0
V22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
V23	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
V24	0.0	0.0	0.0	0.0	10.0	0.0	0.0	0.0	6.0	0.0	5.0	0.0	10.0	6.7
V25	0.0	0.0	0.0	0.0	10.0	0.0	0.0	0.0	6.0	0.0	5.0	0.0	10.0	6.7
VBM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

En condiciones aeróbicas, además del flujo de entrada de glucosa, los siguientes flujos tienen que estar encendidos necesariamente para maximizar cada tipo de fermentación:

- Para maximizar LAC: V02
- Para maximizar ETOH: V02, V03
- Para maximizar AC: V02
- Para maximizar FOR: V02
- Para maximizar aKG: V04, V18, V20
- Para maximizar GLU: V04, V18
- Para maximizar SUCC: V18

En condiciones anaeróbicas, además del flujo de entrada de glucosa, los siguientes flujos tienen que estar encendidos necesariamente para maximizar cada tipo de fermentación:

- Para maximizar LAC: V02
- Para maximizar ETOH: V02, V03
- Para maximizar AC: V11

- Para maximizar FOR: V02, V09, V10
- Para maximizar aKG: V04, V09, V11, V18, V20
- Para maximizar GLU: V04, V09, V11, V18
- Para maximizar SUCC: V18, V24, V25

5.2.5.2.3. Consumo máximo de oxígeno y de ATP

Utilizando los modelos en GLPK, se estudia el consumo de oxígeno y consumo de ATP en ambos modelos (núcleo compacto de E.coli y núcleo original). Como función objetivo para el consumo de oxígeno en el modelo núcleo compacto de E.coli, se usó $V22 + {}^{1}/_{2}\,V23$, ya que la entrada de oxígeno no está presente como flujo en este modelo. En el modelo núcleo original, por el contrario, la entrada de oxígeno sí está presente como "EX_o2(e)" (O2 exchange). Como función objetivo para el consumo de ATP en el modelo núcleo compacto de E.coli, se usó $V19$, que transforma ATP en ADP. En el modelo núcleo original se usó la reacción ATPM:

atp + h2o --> adp + h + pi

Los resultados son los siguientes:

Tabla 27 – Comparación de resultados

Flujo	Máximo compact core	Máximo core original
O2	60	60
ATP (conds. aeróbicas)	175	175
ATP (conds. anaeróbicas)	30	27.5

Se observan resultados muy similares para ambos modelos.

5.3. Cálculos de volumen en modelo núcleo compacto de E.coli

En esta sección se aplican los métodos presentados en la sección 4.4 al modelo núcleo compacto de E.coli. A efectos de poder comparar estos métodos con el método de Monte Carlo directo, se considera un modelo núcleo compacto de E.coli sin fermentaciones, de forma de reducir la cantidad de dimensiones del espacio de soluciones. El método de Monte Carlo directo se usa como referencia y los valores obtenidos se presentan junto con los de los demás métodos.

5.3.1. Muestreo por importancia

Se procede a aplicar muestreo por importancia en el cálculo del volumen del espacio de soluciones (ver 4.4.2) del modelo núcleo compacto de E.coli con una primera simplificación que consiste en eliminar del mismo todas las fermentaciones y también la reacción $V24$, eliminando así el ciclo fútil del modelo, y teniendo un espacio de soluciones en 4 dimensiones. Esta simplificación se hace para poder comparar los

resultados de aplicar un Monte Carlo directo con el uso de muestreo por importancia empleando la distribución triangular así como la exponencial truncada. El objetivo de esto es estudiar la posibilidad de llegar a una estimación satisfactoria al aplicar esta técnica en modelos metabólicos, ya que esto no se consigue mediante el método Monte Carlo directo en modelos de muchas dimensiones. Es importante tener en cuenta que los resultados obtenidos para el modelo ramificado simple no indican que necesariamente se dé una mejora en las estimaciones al aplicar esta técnica.

5.3.1.1. Generación de puntos medios

En un principio, se piensa en aplicar la misma técnica que en la sección 5.1.4.1 para generar un punto medio para utilizar en las distribuciones triangular y exponencial truncada. Sin embargo, al calcular todos los puntos que surgen de maximizar y minimizar cada dimensión en el modelo núcleo compacto de E.coli, se observa que el flujo $V01$ (entrada de glucosa) está en su valor máximo de 10 para todos los puntos. La razón por la que esto sucede es porque el valor de $V01$ en 10 le da más posibilidades de flujo al modelo, permitiéndole alcanzar el máximo o mínimo en cada dimensión. Teniendo esto en cuenta, se decide calcular un punto adicional que surge de minimizar la entrada de glucosa, y a partir de esta información generar dos puntos medios:

- Un punto medio que surge de promediar: los puntos de maximización y minimización de las direcciones y el punto de minimización de $V01$
- Un punto medio que surge de promediar: el promedio de los puntos de maximización y minimización de las direcciones con el punto de minimización de $V01$

Se observa que, en el primer punto medio, se le da menos peso al punto de minimización de glucosa que en el segundo punto medio.

5.3.1.2. Resultados

Los resultados obtenidos con Monte Carlo directo y con muestreo por importancia usando la distribución triangular y la distribución exponencial truncada se presentan a continuación. Los volúmenes se encuentran expresados en u^4. Para todos los experimentos, el volumen de la caja de muestreo es de 9.85E+08. En primer lugar, se usa Monte Carlo directo con dos semillas distintas, obteniéndose los resultados que se presentan en la Tabla 28 y en la Tabla 29.

Tabla 28 – Resultados Monte Carlo directo, semilla = 123

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06	1.00E+07
Cantidad de hits	0	1	19	209	2096	20896	208059
Estimador	0.00E+00	1.00E-02	1.90E-02	2.09E-02	2.10E-02	2.09E-02	2.08E-02
Volumen del espacio de soluciones	0.00E+00	9.85E+06	1.87E+07	2.06E+07	2.07E+07	2.06E+07	2.05E+07
Desviación estandar	0.00E+00	9.85E+06	4.26E+06	1.41E+06	4.46E+05	1.41E+05	4.45E+04
Tiempo transcurrido (s)	0	0.01	0.03	0.16	1.62	16.15	166.99

Tabla 29 – Resultados Monte Carlo directo, semilla = 876

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06	1.00E+07
Cantidad de hits	0	1	16	185	2107	20823	207054
Estimador	0.00E+00	1.00E-02	1.60E-02	1.85E-02	2.11E-02	2.08E-02	2.07E-02
Volumen del espacio de soluciones	0.00E+00	9.85E+06	1.58E+07	1.82E+07	2.08E+07	2.05E+07	2.04E+07
Desviación estandar	0.00E+00	9.85E+06	3.91E+06	1.33E+06	4.48E+05	1.41E+05	4.44E+04
Tiempo transcurrido (s)	0	0	0.02	0.16	1.57	15.41	158.41

Se puede observar cómo se llega a un valor de volumen del espacio de soluciones similar para ambas semillas, y que las estimaciones no varían significativamente a partir de las 10^5 iteraciones. Posteriormente, se aplica muestreo por importancia con la distribución triangular para los dos puntos medios mencionados en la sección 5.3.1.1. Los resultados se presentan en la Tabla 30 y en la Tabla 31.

Tabla 30 – Resultados distribución triangular con primer punto medio (menos peso al punto de minimización de glucosa)

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06	1.00E+07
Cantidad de hits	0	9	114	1058	10805	108285	1081940
Estimador	0.00E+00	1.67E-02	2.03E-02	1.98E-02	2.01E-02	2.02E-02	2.02E-02
Volumen del espacio de soluciones	0.00E+00	1.65E+07	2.00E+07	1.95E+07	1.98E+07	1.99E+07	1.99E+07
Desviación estandar	0.00E+00	5.50E+06	1.98E+06	6.69E+05	2.07E+05	6.68E+04	2.11E+04
Tiempo transcurrido (s)	0	0.01	0.07	0.49	4.97	40.7	422.11

Tabla 31 – Resultados distribución triangular con segundo punto medio (más peso al punto de minimización de glucosa)

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06	1.00E+07
Cantidad de hits	1	16	165	1606	16064	159450	1592919
Estimador	2.35E-02	3.12E-02	2.79E-02	2.63E-02	2.60E-02	2.57E-02	2.57E-02
Volumen del espacio de soluciones	2.31E+07	3.08E+07	2.75E+07	2.59E+07	2.56E+07	2.53E+07	2.53E+07
Desviación estandar	2.31E+07	8.03E+06	2.24E+06	6.71E+05	2.07E+05	6.70E+04	2.11E+04
Tiempo transcurrido (s)	0	0.01	0.07	0.43	5.2	44.83	395.14

Se puede observar cómo, en este caso, las estimaciones se estabilizan pero no coinciden los resultados para ambos puntos medios. Finalmente, se aplica muestreo por importancia con la distribución exponencial para el primer punto medio para distintos valores de λ.

Tabla 32 – Resultados distribución exponencial truncada con $\lambda = 0.005$

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06
Cantidad de hits	1	3	43	413	3971	39265
Estimador	7.20E-02	1.81E-02	2.24E-02	2.19E-02	2.11E-02	2.09E-02
Volumen del espacio de soluciones	7.10E+07	1.78E+07	2.20E+07	2.16E+07	2.08E+07	2.06E+07
Desviación estandar	7.10E+07	1.05E+07	3.41E+06	1.09E+06	3.42E+05	1.08E+05
Tiempo transcurrido (s)	0	0.01	0.07	0.64	6.51	62.73

Tabla 33 – Resultados distribución exponencial truncada con λ = 0.01

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06
Cantidad de hits	1	3	51	539	5030	50421
Estimador	4.14E-02	1.11E-02	1.87E-02	2.17E-02	2.06E-02	2.08E-02
Volumen del espacio de soluciones	4.08E+07	1.09E+07	1.85E+07	2.14E+07	2.03E+07	2.05E+07
Desviación estandar	4.08E+07	6.31E+06	2.66E+06	9.88E+05	3.08E+05	9.81E+04
Tiempo transcurrido (s)	0	0.01	0.07	0.63	6.65	66.59

Tabla 34 – Resultados distribución exponencial truncada con λ = 0.1

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06
Cantidad de hits	1	27	320	3355	33582	333453
Estimador	7.95E-05	1.34E-02	2.99E-02	1.20E-02	1.69E-02	1.91E-02
Volumen del espacio de soluciones	7.83E+04	1.32E+07	2.94E+07	1.18E+07	1.67E+07	1.88E+07
Desviación estandar	7.83E+04	1.18E+07	1.86E+07	2.20E+06	1.99E+06	1.52E+06
Tiempo transcurrido (s)	0	0.01	0.08	0.75	7.48	75.28

Tabla 35 – Resultados distribución exponencial truncada con λ = 0.2

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06
Cantidad de hits	4	46	459	4650	46261	462430
Estimador	5.94E-04	3.31E-03	4.86E-03	2.89E-03	1.03E-02	6.01E-02
Volumen del espacio de soluciones	5.85E+05	3.27E+06	4.79E+06	2.85E+06	1.02E+07	5.92E+07
Desviación estandar	3.64E+05	2.25E+06	2.16E+06	4.56E+05	3.70E+06	5.10E+07
Tiempo transcurrido (s)	0	0.01	0.08	0.74	7.53	73.14

Tabla 36 – Resultados distribución exponencial truncada con λ = 0.5

Iteraciones	1.00E+01	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06
Cantidad de hits	4	51	501	5038	50045	500579
Estimador	1.52E-05	1.38E-04	2.12E-04	5.80E-04	7.02E-04	1.91E-03
Volumen del espacio de soluciones	1.50E+04	1.36E+05	2.09E+05	5.72E+05	6.92E+05	1.89E+06
Desviación estandar	9.31E+03	6.43E+04	7.69E+04	2.97E+05	2.10E+05	1.31E+06
Tiempo transcurrido (s)	0	0.01	0.08	0.74	7.52	73.84

Se puede observar cómo, para los valores más bajos de λ, los resultados que se obtienen son similares a los obtenidos para Monte Carlo directo, al asemejarse la distribución exponencial truncada a una distribución uniforme. Sin embargo, para los valores más altos de λ, no se llega a un resultado estable en 10^6 iteraciones.

5.3.1.3. Conclusiones del uso de muestreo por importancia

A modo de conclusión de esta etapa, se observa que el uso de muestreo por importancia para el cálculo del volumen del espacio de soluciones de un modelo metabólico no da resultados satisfactorios. Esto puede deberse, por un lado, a la dificultad de generar un punto medio adecuado para el uso de las distribuciones alternativas. Una mejor metodología para generar dicho punto medio (por ejemplo, intentando usar un Monte

Carlo directo para generar un punto medio) podría implicar una mejora en la implementación de muestreo por importancia. Por otro lado, también es necesario tener en cuenta que, en los modelos de mayor tamaño, pueden existir correlaciones entre las dimensiones que no necesariamente son capturadas con muestreo por importancia de la forma en que está implementado en este trabajo (con distribuciones independientes en cada dimensión).

Analizándolo desde el punto de vista geométrico, según lo mencionado en la sección 4.4.3, los espacios de soluciones de modelos metabólicos suelen ser de una forma particular: convexos pero con una forma muy irregular (alargados para las reacciones que no están muy restringidas, y angostos para los flujos más restringidos). Por ello, si bien el uso de muestreo por importancia intenta concentrar las muestras alrededor de un punto medio interior conocido, esto no asegura que las muestras relativamente cercanas a ese punto queden adentro del volumen.

Finalmente, es interesante ver los histogramas que se producen al aplicar Monte Carlo directo al modelo núcleo compacto de E.coli sin fermentaciones, en cada una de las 5 dimensiones de muestreo correspondientes. Las siguientes ilustraciones presentan estos resultados para 10^6 iteraciones. Se puede observar cómo, en cada dimensión, las distribuciones de las muestras parecen asemejarse más a una distribución triangular o una exponencial truncada que a una distribución uniforme. Esto lleva, intuitivamente, a pensar que el uso de muestreo por importancia podría dar buenos resultados en la resolución del problema del cálculo de volumen del espacio de soluciones, aunque en la práctica esto no necesariamente se dé así, especialmente teniendo en cuenta que la generación del punto medio no es un problema trivial. Asimismo, en el caso de dimensiones más altas, las distribuciones pueden no asemejarse al caso graficado.

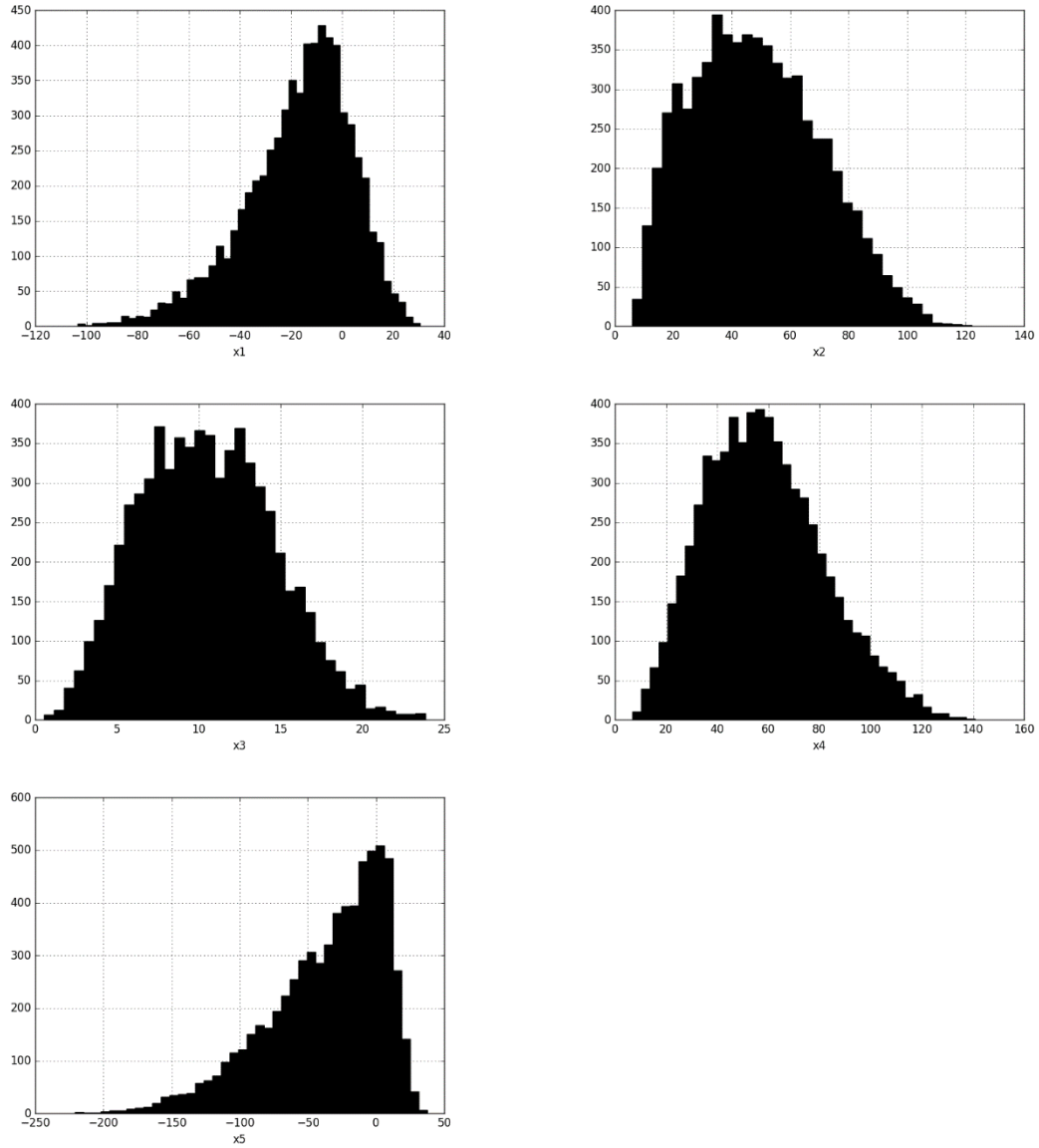


Ilustración 31 – Histogramas de Monte Carlo directo en modelo núcleo compacto de E.coli sin fermentaciones

5.3.2. Uso de COBRApy sampler

5.3.2.1. Introducción

Se aplica la misma metodología que se usó para el modelo ramificado simple en la sección 5.1.3 al modelo núcleo compacto de E.coli, en este caso sin fermentaciones. El modelo con esta particularidad (fermentaciones apagadas) tiene un total de 18 reacciones y 13 metabolitos independientes, con un total de 5 dimensiones del espacio de soluciones. El objetivo es calcular el volumen haciendo uso del sampler de COBRApy para empezar por el volumen de la caja de muestreo e irse aproximando paso a paso (agregando una restricción de flujo por vez) al volumen del espacio de soluciones. Ya que las dimensiones de este modelo son relativamente bajas, es posible hacer una comparación entre este método y la aplicación de un Monte Carlo directo en la caja de muestreo. El resultado de aplicar Monte Carlo directo con 10^7 iteraciones al modelo planteado da un volumen de 1.87E+08 u^5. Este resultado se plantea también en la sección 5.4.1, donde, entre otros, se calcula el volumen para el modelo sin fermentaciones.

5.3.2.2. Procedimiento

Se aplican de a una las 36 restricciones de flujo (una de LB y una de UB para cada reacción), haciendo en primer lugar un muestreo para la caja entera y después 36 muestreos sucesivos agregando una a una las restricciones. En total, se tiene un total de 37 muestreos y 36 fracciones (en base a cuyo producto se encuentra el estimador del volumen del espacio de soluciones). Se empieza por agregar todas las restricciones de LB y después las de UB. Las restricciones se listan en la Tabla 37.

Tabla 37 – Restricciones

	LB	UB
V01	0	10
V02	0	1000
V03	0	1000
V04	0	1000
V05	-1000	1000
V06	0	1000
V07	0	1000
V14	0	1000
V15	0	20
V16	-1000	1000
V18	0	1000
V19	8.39	1000
V20	-1000	1000
V22	0	1000
V23	0	1000
V24	0	1000
V25	0	1000
VBM	0	1000

A su vez, la caja de muestreo determinada por GLPK es la siguiente, y tiene un volumen de 2.75E+10.

Tabla 38 – Caja de muestreo

	cota inf	cota sup
x1	-131.57	33.16
x2	1.65	128.60
x3	-0.30	27.35
x4	4.18	154.33
x5	-271.67	45.30

5.3.2.3. Resultados

Se aplica la metodología con 10^2, 10^3, 10^4 y 10^5 iteraciones. Los resultados se listan en la Tabla 39.

Tabla 39 – Resultados COBRApy para núcleo compacto de E.coli

Cantidad de muestras	Tiempo total de muestreo (segundos)	Estimador	Volumen estimado del espacio de soluciones	% diferencia con MC directo
100	36.76	0.010867	2.99E+08	59.56
1000	363.22	0.010355	2.85E+08	52.03
10000	3101.64	0.010289	2.83E+08	51.07
100000	34700.30	0.008874	2.44E+08	30.29

Dados los resultados de la aplicación de Monte Carlo directo y del método con COBRApy para cálculo del volumen del espacio de soluciones en el modelo ramificado simple, se entiende que los resultados del Monte Carlo directo para el núcleo compacto de E.coli son más confiables que los obtenidos con COBRApy. Observando los porcentajes de diferencia entre un método y otro, se puede concluir que el método con COBRApy no da una buena aproximación pero permite llegar al orden del volumen del espacio de soluciones, lo que puede ser de utilidad en casos de mayores dimensiones en los que un método Monte Carlo directo no pueda generar una estimación.

Una observación interesante es que este método permite ver cuál es la contribución parcial de cada restricción a la reducción en el volumen del espacio de soluciones, pasando desde el volumen inicial que corresponde a la caja de muestreo al volumen final que corresponde al espacio de soluciones del sistema. Es decir, para cada restricción que se agrega, puede verse cómo esta modifica el volumen resultante. La Ilustración 32 presenta, para cada restricción (LB y UB), la fracción correspondiente encontrada en el caso de las 10^5 iteraciones. Lo que esto representa es el porcentaje del volumen del paso anterior que cumple cada restricción (tanto de LB como de UB). Es importante recordar que, en estos experimentos, se implementaron primero todas las reacciones de LB y después todas las de UB, empezando por $V01$ y terminando por VBM.

	V01	V02	V03	V04	V05	V06	V07	V14	V15	V16	V18	V19	V20	V22	V23	V24	V25	VBM
LB	0,62	1,00	1,00	1,00	1,00	0,98	1,00	1,00	1,00	1,00	0,51	0,82	1,00	0,91	0,99	0,73	0,88	1,00
UB	0,14	1,00	1,00	1,00	1,00	1,00	1,00	1,00	0,42	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00

Ilustración 32 – Fracciones para 10^5 iteraciones

Se puede observar que la restricción más significativa es la de UB de $V01$, que corresponde a la limitación de entrada de glucosa en 10, consistente con lo esperable ya que esta entrada de nutriente es indispensable para el funcionamiento del sistema.

También $V15$ y $V19$, en las que hemos limitado la cota superior e inferior, respectivamente, corresponden a restricciones significativas. Es importante notar que, en este caso, ninguna restricción hace que todas las muestras del paso anterior queden afuera del espacio. En otras palabras, no hay ninguna restricción que no sea cumplida por al menos algunas de las muestras del paso anterior (en cuyo caso la fracción sería 0, y no sería posible estimar un volumen final). Si ese fuera el caso, sería necesario hacer una mayor cantidad de pasos, implementando la restricción de a poco. Por ejemplo, si un UB de 10 para una reacción diera como resultado una fracción de 0, podría implementarse primero un UB de 100, luego 50 y finalmente 10, de manera de no tener un salto tan grande entre un espacio y otro.

5.4. Análisis de volúmenes del núcleo compacto de E.coli

5.4.1. Apagando fermentaciones

5.4.1.1. Introducción

El modelo núcleo compacto de E.coli cuenta con 7 fermentaciones: LAC, ETOH, AC, FOR, aKG, GLU, SUCC. Utilizando Monte Carlo directo en un espacio de referencia construido con descomposición QR, se procede a hacer el cálculo de volumen del espacio de soluciones en las siguientes situaciones:

- Una situación sin fermentaciones (UB=0 para todas las fermentaciones)
- 7 situaciones prendiendo una de las fermentaciones cada vez
- 21 situaciones prendiendo dos fermentaciones a la vez

Cabe destacar que el espacio de referencia debe ser construido para cada una de las situaciones. En cada situación, se eliminan del modelo los flujos correspondientes y se aplica el procedimiento explicado en la sección 4.4.1.2.1 para encontrar la base ortonormal del espacio de soluciones y las cotas superiores e inferiores de la caja. El código para armar las cajas de muestreo se presenta en el Anexo 11.

5.4.1.2. Identificación de "sinergias" entre fermentaciones

Se define el factor $f(i)$ a partir de la ecuación: $V(i) = f(i) * V_{resp}$ con $i = 1,\ldots,7$. V_{resp} es el volumen obtenido con todas las fermentaciones apagadas y $V(i)$ es el volumen obtenido cuando solamente la fermentación i está prendida.

Similarmente, se define el factor $f(i,j)$ a partir de la ecuación: $V(i,j) = f(i,j) * V_{resp}$ con $i = 1,\ldots,7$; $j = 1,\ldots,7$; i distinto de j. De nuevo, V_{resp} es el volumen obtenido con todas las fermentaciones apagadas y $V(i,j)$ es el volumen obtenido cuando solamente las fermentaciones i y j están prendidas.

Se propone comparar $f(i,j)$ con el producto $f(i) * f(j)$. Si son iguales, los efectos serían independientes. Si $f(i,j) > f(i) * f(j)$ habría sinergia positiva entre las

fermentaciones i y j. Si $f(i,j) < f(i) * f(j)$ habría sinergia negativa entre las fermentaciones i y j.

5.4.1.3. Resultados

Se utilizó un total de 10^7 iteraciones para cada una de las situaciones. La Tabla 40 presenta los resultados obtenidos, donde los volúmenes se encuentran en distintas dimensiones según la cantidad de fermentaciones apagadas en el modelo.

Tabla 40 – Resultados

Fermentaciones encendidas		Vol caja	Vol esp sol	Desv est
ninguna fermentación		2.75E+10	1.87E+08	7.16E+05
V09	V10	6.49E+14	1.98E+10	1.13E+09
V10		8.72E+12	3.54E+09	5.56E+07
V10	V17	3.42E+14	2.65E+10	9.52E+08
V10	V13	7.50E+14	1.73E+10	1.14E+09
V10	V12	6.40E+14	1.89E+10	1.10E+09
V10	V11	4.46E+14	4.96E+10	1.49E+09
V17		1.02E+13	2.21E+09	4.75E+07
V13		7.46E+12	1.53E+09	3.38E+07
V13	V17	6.13E+14	9.93E+09	7.80E+08
V12		9.01E+12	1.80E+09	4.03E+07
V12	V17	3.58E+14	8.85E+09	5.63E+08
V12	V13	1.97E+14	7.64E+09	3.88E+08
V11		1.27E+13	2.62E+09	5.77E+07
V11	V17	8.20E+14	2.67E+10	1.48E+09
V11	V13	7.79E+14	1.96E+10	1.24E+09
V11	V12	7.75E+14	2.07E+10	1.27E+09
V09		2.24E+13	1.77E+09	6.30E+07
V09	V17	1.38E+15	1.41E+10	1.39E+09
V09	V13	1.65E+15	9.89E+09	1.28E+09
V09	V12	1.60E+15	1.12E+10	1.34E+09
V09	V11	5.42E+14	2.35E+10	1.13E+09
V08	V10	6.18E+14	1.60E+10	9.93E+08
V08		1.07E+13	1.44E+09	3.92E+07
V08	V17	6.40E+14	1.15E+10	8.59E+08
V08	V13	5.50E+14	7.86E+09	6.57E+08
V08	V12	5.76E+14	8.75E+09	7.10E+08
V08	V11	4.94E+14	1.52E+10	8.67E+08
V08	V09	8.76E+14	8.93E+09	8.84E+08

Según se puede observar, se encontró un volumen de 2.75×10^{10} para la situación sin ninguna fermentación prendida, el cual representa V_{resp}. Se procede a hacer los cálculos de sinergia entre las fermentaciones. Los resultados se presentan en la Tabla 41.

Tabla 41 – Sinergia

rxns encendidas		f(i,j)	f(i)	f(j)	f(i)*f(j)	sinergia
V09	V10	105.59	9.47	18.91	179.05	-
V10	V17	141.35	18.91	11.79	222.94	-
V10	V13	92.45	18.91	8.17	154.40	-
V10	V12	101.08	18.91	9.61	181.73	-
V10	V11	264.83	18.91	13.96	263.94	+
V13	V17	52.96	8.17	11.79	96.29	-
V12	V17	47.24	9.61	11.79	113.34	-
V12	V13	40.75	9.61	8.17	78.49	-
V11	V17	142.57	13.96	11.79	164.61	-
V11	V13	104.67	13.96	8.17	114.00	-
V11	V12	110.44	13.96	9.61	134.19	-
V09	V17	74.99	9.47	11.79	111.67	-
V09	V13	52.75	9.47	8.17	77.33	-
V09	V12	59.66	9.47	9.61	91.03	-
V09	V11	125.44	9.47	13.96	132.20	-
V08	V10	85.12	7.67	18.91	145.04	-
V08	V17	61.46	7.67	11.79	90.46	-
V08	V13	41.95	7.67	8.17	62.65	-
V08	V12	46.70	7.67	9.61	73.74	-
V08	V11	81.16	7.67	13.96	107.10	-
V08	V09	47.65	7.67	9.47	72.65	-

Se observa que la sinergia es siempre negativa, con la excepción de un solo caso en el que los valores de $f(i,j)$ y $f(i) * f(j)$ son demasiado cercanos como para poder afirmar la existencia de una sinergia positiva (teniendo en cuenta las desviaciones estándar de los valores).

5.4.2. Fijando fermentaciones

5.4.2.1. Introducción

Se tiene el volumen donde el organismo solamente respira (con todas las fermentaciones apagadas), el cual tiene 5 dimensiones. Surge la pregunta de cómo cambia ese espacio de 5 dimensiones al agregar un producto de fermentación. El espacio de 5 dimensiones original se puede considerar como el volumen obtenido para velocidad de fermentación cero. Ahora, si la fermentación no vale cero, ¿cuánto pasa a valer el volumen donde el organismo solo respira, de 5 dimensiones? Más aún, ¿para qué valor de la velocidad de fermentación el volumen correspondiente a solo respirar es máximo?

La existencia de respiro-fermentación sigue siendo un fenómeno debatido, o sea, ¿por qué en presencia de oxígeno el organismo respiro-fermenta en lugar de solo respirar? Una explicación posible es que al no hacer nula la velocidad de una fermentación, podría aumentar la variabilidad de la respiración, incrementando la capacidad adaptativa del organismo. Para analizar la plausibilidad de esta hipótesis, se procede a estudiar el efecto de las fermentaciones sobre el volumen de la respiración en el modelo núcleo compacto de E.coli.

5.4.2.2. Procedimiento

Se apagan todas las fermentaciones, y se las va habilitando de a una a la vez. Esto resulta en 7 casos, uno para cada fermentación. En cada caso se hacen 11 corridas, que corresponden a fijar la velocidad de fermentación en: Vf_{max} (velocidad máxima que puede adoptar esa fermentación), $0.9 * Vf_{max}, \ldots, 0.1 * Vf_{max}$ y cero (el valor cero se usa para verificar que hacer la velocidad de fermentación cero y no incorporar esta velocidad en la matriz dan el mismo resultado). Como resultado se tienen 11 volúmenes de la respiración, para 11 valores fijos diferentes de la velocidad de cada fermentación.

Los valores de Vf_{max} se calculan mediante GLPK con las restricciones estequiométricas y las de LB y UB de flujo, maximizando la fermentación correspondiente para un modelo con solo esa fermentación prendida (es decir, UB de todas las demás fermentaciones en cero). Los valores obtenidos se presentan en la Tabla 42 (consistentemente con lo presentado en la sección 5.2.5.2.2).

Tabla 42 – Flujos máximos para las fermentaciones

fermentación	vfmax
V08	20
V09	20
V10	20
V11	20
V12	10
V13	10
V17	10

La metodología es análoga a la explicada en la sección 5.1.5 para el modelo ramificado simple. En este caso, las soluciones x_0 se generan utilizando modelos en GLPK adaptados a cada situación, maximizando una variable cualquiera para tener una solución factible. El pseudocódigo se describe a continuación:

```
maximizar v[1]
s.a:
    • balance{j in 1..13}: sum{i in 1..18} S[j,i]*v[i]=b[j]
    • upperbound{i in 1..18}: v[i] <= Vmax[i]
    • lowerbound{i in 1..18}: v[i] >= Vmin[i]
```

v[1] puede ser cualquiera de los flujos (cambiando por otro se generaría un x_0 diferente pero también válido). S[j,i] es la matriz estequiométrica sin ninguna de las fermentaciones (teniendo en cuenta que esta se aplica a todas las situaciones que se tratan en este apartado, porque siempre se eliminan del modelo 6 de las 7 fermentaciones y la que no se elimina se deja fija, por lo que tampoco está presente en la matriz estequiométrica modificada).

Una vez obtenidas las soluciones x_0 correspondientes a cada caso, se procede a generar las cajas de muestreo. Las cajas de muestreo se generan mediante el método de la descomposición QR. Se aplica el método de Monte Carlo directo para 10^7 replicaciones. Los resultados se presentan a continuación.

5.4.2.3. Resultados Monte Carlo

La Tabla 43 presenta los resultados obtenidos.

Tabla 43 – Resultados

flujo fijado	valor fijado al % de vfmax	Vol caja	Vol espacio de respiración (u^5)	desv est
V08	0%	2.75E+10	1.87E+08	7.16E+05
V08	10%	1.86E+10	1.53E+08	5.31E+05
V08	20%	1.21E+10	1.15E+08	3.71E+05
V08	30%	7.41E+09	8.04E+07	2.43E+05
V08	40%	4.23E+09	5.13E+07	1.47E+05
V08	50%	2.18E+09	2.95E+07	7.98E+04
V08	60%	9.84E+08	1.46E+07	3.76E+04
V08	70%	3.62E+08	5.83E+06	1.44E+04
V08	80%	9.51E+07	1.60E+06	3.87E+03
V08	90%	1.23E+07	1.82E+05	4.69E+02
V08	100%	2.72E+04	0.00E+00	0.00E+00
V09	0%	2.75E+10	1.87E+08	7.16E+05
V09	10%	1.86E+10	1.53E+08	5.31E+05
V09	20%	1.21E+10	1.15E+08	3.71E+05
V09	30%	7.41E+09	8.04E+07	2.43E+05
V09	40%	4.23E+09	5.13E+07	1.47E+05
V09	50%	2.18E+09	2.95E+07	7.98E+04
V09	60%	9.84E+08	1.46E+07	3.76E+04
V09	70%	3.62E+08	5.83E+06	1.44E+04
V09	80%	9.51E+07	1.60E+06	3.87E+03
V09	90%	1.23E+07	1.82E+05	4.69E+02
V09	100%	2.72E+04	0.00E+00	0.00E+00
V10	0%	2.75E+10	1.87E+08	7.16E+05
V10	10%	2.32E+10	1.89E+08	6.59E+05
V10	20%	1.87E+10	1.80E+08	5.78E+05
V10	30%	1.48E+10	1.64E+08	4.91E+05
V10	40%	1.16E+10	1.43E+08	4.04E+05
V10	50%	8.80E+09	1.19E+08	3.21E+05
V10	60%	6.30E+09	9.13E+07	2.38E+05
V10	70%	4.16E+09	6.11E+07	1.58E+05
V10	80%	2.57E+09	3.19E+07	8.99E+04
V10	90%	1.45E+09	9.25E+06	3.65E+04
V10	100%	7.18E+08	0.00E+00	0.00E+00
V11	0%	2.75E+10	1.87E+08	7.16E+05
V11	10%	2.49E+10	1.73E+08	6.53E+05
V11	20%	2.17E+10	1.59E+08	5.85E+05
V11	30%	1.89E+10	1.44E+08	5.20E+05
V11	40%	1.63E+10	1.28E+08	4.55E+05
V11	50%	1.40E+10	1.08E+08	3.88E+05
V11	60%	1.19E+10	8.41E+07	3.16E+05
V11	70%	9.85E+09	5.48E+07	2.32E+05
V11	80%	6.93E+09	2.45E+07	1.30E+05
V11	90%	4.57E+09	4.31E+06	4.44E+04
V11	100%	2.81E+09	0.00E+00	0.00E+00
V12	0%	2.75E+10	1.87E+08	7.16E+05
V12	10%	2.18E+10	1.76E+08	6.16E+05
V12	20%	1.67E+10	1.55E+08	5.07E+05
V12	30%	1.25E+10	1.30E+08	4.01E+05
V12	40%	9.03E+09	1.04E+08	3.05E+05
V12	50%	6.33E+09	7.90E+07	2.22E+05
V12	60%	4.28E+09	5.58E+07	1.53E+05
V12	70%	2.67E+09	3.53E+07	9.64E+04
V12	80%	1.48E+09	1.80E+07	5.12E+04
V12	90%	7.22E+08	5.13E+06	1.92E+04
V12	100%	2.91E+08	0.00E+00	0.00E+00
V13	0%	2.75E+10	1.87E+08	7.16E+05
V13	10%	2.08E+10	1.69E+08	5.91E+05
V13	20%	1.53E+10	1.43E+08	4.66E+05

V13	30%	1.09E+10	1.14E+08	3.50E+05
V13	40%	7.40E+09	8.59E+07	2.51E+05
V13	50%	4.82E+09	6.09E+07	1.70E+05
V13	60%	2.97E+09	3.99E+07	1.08E+05
V13	70%	1.71E+09	2.34E+07	6.28E+04
V13	80%	8.28E+08	1.12E+07	3.02E+04
V13	90%	3.29E+08	3.04E+06	9.96E+03
V13	100%	9.42E+07	0.00E+00	0.00E+00
V17	0%	2.75E+10	1.87E+08	7.16E+05
V17	10%	2.32E+10	1.88E+08	6.58E+05
V17	20%	1.89E+10	1.78E+08	5.77E+05
V17	30%	1.51E+10	1.61E+08	4.90E+05
V17	40%	1.19E+10	1.40E+08	4.05E+05
V17	50%	9.17E+09	1.17E+08	3.25E+05
V17	60%	6.90E+09	9.19E+07	2.50E+05
V17	70%	5.00E+09	6.62E+07	1.81E+05
V17	80%	3.45E+09	3.96E+07	1.16E+05
V17	90%	1.95E+09	1.36E+07	5.13E+04
V17	100%	9.37E+08	0.00E+00	0.00E+00

De forma de poder visualizar el efecto de la fermentación fijada en el volumen, se construye la gráfica que se presenta en la Ilustración 33.

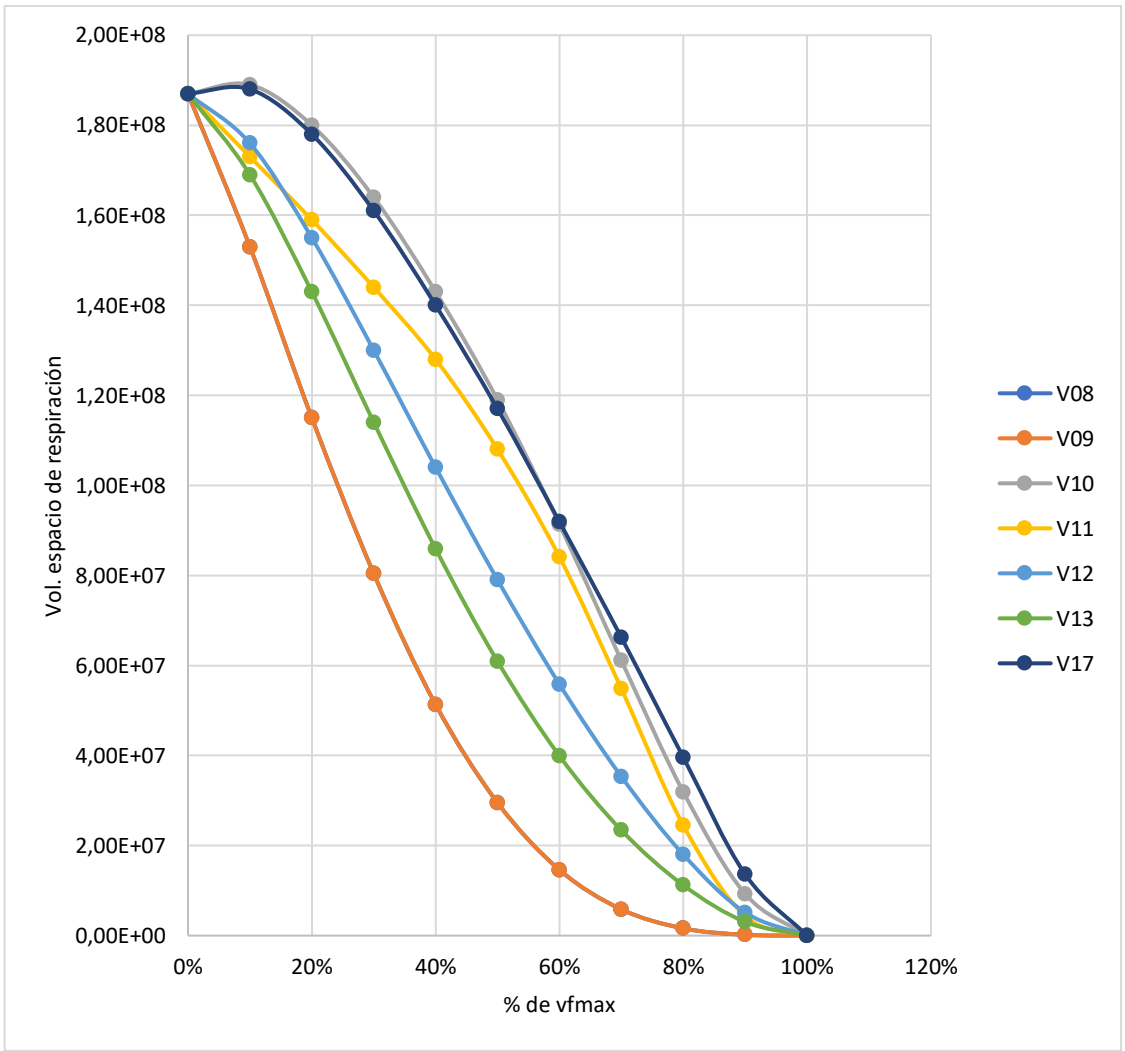


Ilustración 33 – Resultados

Se observa de la gráfica que el hecho de fijar una fermentación en un valor superior a 0 hace que el volumen del espacio de soluciones disminuya (a excepción del comienzo de la gráfica para $V10$ y $V17$, pero las desviaciones estándar indican que esto puede deberse al error de cálculo inherente a la metodología empleada). Esto implica que, en el modelo núcleo compacto de E.coli, forzar el flujo a través de una determinada fermentación no aumenta, al menos significativamente, el volumen de la respiración.

5.4.2.4. Integración de los volúmenes

Al igual que para el modelo ramificado simple, se hace la integración de las secciones encontradas para volver al volumen original. El volumen original en cada uno de los siete casos corresponde al volumen del espacio de soluciones del modelo núcleo compacto de E.coli con la fermentación correspondiente libre (no fija). Es necesario hacer el cálculo de las distancias que se describe en el modelo ramificado simple, que se aplica también a dimensiones más altas. Para cada situación, se toma la matriz W correspondiente a la situación con la fermentación correspondiente libre. De esa matriz W, se usa la fila correspondiente a la fermentación para obtener las ecuaciones de los distintos hiperplanos de corte correspondientes a los distintos valores en los que se fija la fermentación y calcular la distancia entre ellos.

A modo de ejemplo, la matriz W correspondiente a la situación con $V11$ libre (y todas las demás fermentaciones apagadas) aparece en la Tabla 44.

Tabla 44 – Matriz W como ejemplo

V01	-0.193	0.014	0.110	-0.023	0.090	0.132
V02	-0.122	-0.012	0.060	-0.001	0.078	0.079
V03	0.254	0.340	0.426	-0.183	0.137	-0.072
V04	-0.178	-0.048	0.072	0.020	0.144	0.108
V05	0.092	0.075	-0.019	-0.181	-0.110	0.554
V06	-0.206	-0.033	0.100	0.145	0.129	-0.469
V07	-0.201	-0.035	0.096	0.147	0.128	-0.473
V11	-0.510	-0.360	-0.298	0.179	0.021	0.239
V14	-0.155	-0.057	0.056	0.027	0.141	0.091
V15	-0.026	0.240	0.103	0.639	-0.010	0.192
V16	-0.155	-0.057	0.056	0.027	0.141	0.091
V18	-0.060	0.022	0.043	-0.019	0.010	0.045
V19	0.550	-0.626	0.145	0.225	0.361	0.117
V20	0.204	-0.188	-0.198	0.139	0.080	-0.176
V22	-0.096	0.129	-0.047	-0.010	0.430	0.099
V23	-0.275	-0.220	0.763	-0.012	-0.029	0.144
V24	0.129	0.297	0.047	0.613	-0.150	0.101
V25	-0.037	0.315	-0.149	-0.047	0.719	0.108
VBM	-0.021	0.008	0.015	-0.007	0.004	0.016

Se observa que cada una de las filas de esta matriz corresponde a uno de los flujos en el modelo en la situación determinada. Los flujos son el resultado de $Wx = V$, siendo x un vector aleatorio. Por ende, $V11$ puede expresarse como:

$$V11 = -0.510x_1 - 0.360x_2 - 0.298x_3 + 0.179x_4 + 0.021x_5 + 0.239x_6$$

Al decidir fijar $V11$ en un valor b, se tiene la ecuación que describe al hiperplano de corte:

$$-0.510x_1 - 0.360x_2 - 0.298x_3 + 0.179x_4 + 0.021x_5 + 0.239x_6 = b$$

Recordando la fórmula de distancia entre dos hiperplanos presentada en la sección 5.1.5, es necesario conocer la norma euclideana del vector a, que en el caso del ejemplo sería:

$$\|a\|_2 = \sqrt{0.510^2 + 0.360^2 + 0.298^2 + 0.179^2 + 0.021^2 + 0.239^2} = 0.754$$

Los resultados para todas las fermentaciones se presentan en la Tabla 45.

Tabla 45 – Resultados de integración de volúmenes

Flujo	Norma de vector a	Distancia entre cortes	Volumen integrado (u^6)	Volumen con flujo libre (u^6)	% error
V08	0.79	2.54	1.39E+09	1.44E+09	3.65%
V09	0.64	3.13	1.71E+09	1.77E+09	3.90%
V10	0.61	3.26	3.53E+09	3.54E+09	0.32%
V11	0.75	2.65	2.58E+09	2.62E+09	1.25%
V12	0.49	2.05	1.75E+09	1.80E+09	3.11%
V13	0.49	2.04	1.51E+09	1.53E+09	1.05%
V17	0.50	2.00	2.17E+09	2.21E+09	1.76%

Se puede observar que los volúmenes calculados a partir de los cortes se acercan a los obtenidos previamente en la sección 5.4, donde las fermentaciones estaban libres. Con una mayor cantidad de secciones, las aproximaciones mejorarían.

5.4.2.5. FVA en fermentaciones

Como último paso en el análisis, se procede a hacer FVA para ver cuáles flujos (y en qué medida) aumentan su variabilidad al activarse las distintas fermentaciones. Se toman 6 valores para cada fermentación (cero, máximo y cuatro intermedios). Para cada uno de estos valores se hace un análisis FVA del modelo núcleo compacto de E.coli, por una parte fijando el flujo de la fermentación en ese valor y por otra parte tomando ese valor como la cota superior del flujo de la fermentación (con las demás fermentaciones apagadas). La idea es determinar cuáles son los principales flujos responsables de los cambios en el volumen que se determinaron anteriormente.

Una vez obtenidos los resultados del FVA, se dividen los máximos y mínimos obtenidos con las fermentaciones prendidas entre los valores correspondientes con las fermentaciones apagadas. Además, se calculan las variabilidades de los flujos (máximo-mínimo) expresándolas también en términos absolutos y relativos a los valores correspondientes con las fermentaciones apagadas.

A modo de ejemplo de los resultados obtenidos, se presenta la siguiente gráfica correspondiente a fijar $V09$ en un valor de $0.2 * Vf_{max}$. Las barras azules de la ilustración representan la variabilidad de cada flujo, es decir, el resultado de dividir la diferencia entre capacidad máxima y mínima para esta situación por la diferencia entre

capacidad máxima y mínima para la referencia (caso sin fermentaciones). Se observa cómo la variabilidad está por debajo de 1 para la mayoría de los flujos.

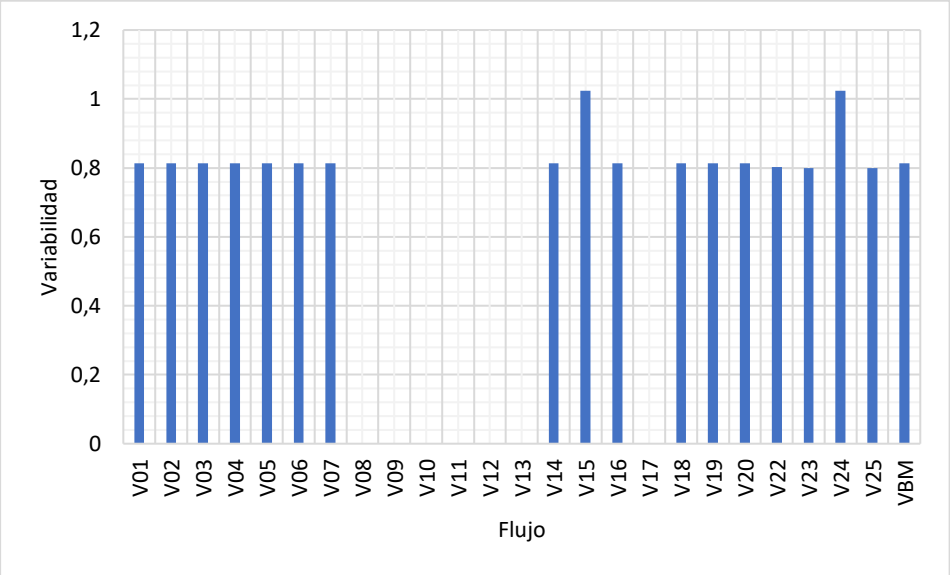


Ilustración 34 – Ejemplo de resultados FVA

5.4.3. Restringiendo la suma de los valores absolutos de los flujos

5.4.3.1. Introducción

A efectos de incorporar en las simulaciones el hecho de que existe una capacidad máxima de proteína que la célula puede acumular en su volumen, se agrega una restricción adicional para el modelo que corresponde a acotar la sumatoria de los valores absolutos de todos los flujos [25]. Es decir, una restricción del tipo:

$$\sum_{i=1}^{n} |v_i| \leq \alpha$$

Es interesante analizar distintos escenarios con distintos valores de la constante α, significando un aumento o disminución en la capacidad máxima de la célula. Para determinar estos valores, es conveniente conocer cuánto es el máximo y mínimo valor que la sumatoria de valores absolutos de los flujos puede tomar en el modelo. Por lo tanto, es necesario resolver el problema de optimización que permite encontrar estos valores máximo y mínimo.

Se plantea un modelo en GLPK para encontrar dichos valores, donde la función objetivo corresponde a la suma de valores absolutos de los flujos. Como restricciones en el modelo se tienen las restricciones estequiométricas y las restricciones de capacidad máxima y mínima de cada flujo. De manera de poder modelar el valor absoluto de cada flujo, es necesario pasar de un problema de programación lineal a un problema de programación lineal entera. Es decir, corresponde definir una variable binaria para cada flujo que permita modelar su valor absoluto, según se explica a continuación.

Teniendo una variable x, cuyo valor absoluto $|x|$ se desea maximizar o minimizar, se agregan al programa las siguientes cuatro restricciones:

1. $x - MV \leq u \leq x + MV$
2. $-x - M(1 - V) \leq u \leq -x + M(1 - V)$
3. $x \leq M(1 - V)$
4. $-MV \leq x$

donde u es una variable continua, V es una variable binaria $\{0,1\}$ y M es un número arbitrariamente grande. Sustituyo $|x|$ en la función objetivo por u, y procedo a maximizar/minimizar. La restricción 1 asegura que, si $V = 0$, $u = x$. La restricción 2 asegura que, si $V = 1$, $u = -x$. La restricción 3 asegura que, si $x > 0$, $V = 0$. La restricción 4 asegura que, si $x < 0$, $V = 1$. Por lo tanto, u siempre es igual a $|x|$.

Cabe destacar que, en la práctica usual de la programación matemática, se distingue (y modela distinto) el caso de minimizar valores absolutos (que se puede resolver sin incorporar variables binarias, alcanza solo con expresar la variable x como la resta de dos subvariables positivas, $x = x^+ - x^-$, y luego minimizar $(x^+ + x^-)$) del caso en que se quiere maximizar valores absolutos, que se puede resolver incorporando una nueva variable continua y una variable binaria.

La propuesta, planteada más arriba, funciona bien en ambos casos, si bien implica usar siempre variables binarias, lo que aumenta el tiempo necesario para la resolución. El caso de minimización podría resolverse sin recurrir a las mismas. En la medida que para los problemas que se estudian en este trabajo el tiempo de resolución no implica un problema, se usa siempre el procedimiento de programación lineal entera presentado.

Se busca el máximo y mínimo de la sumatoria de valores absolutos de los flujos para el modelo núcleo compacto de E.coli en las situaciones: sin fermentaciones y prendiéndolas de a una por vez, tanto para el caso con producción de biomasa como para el caso sin producción de biomasa (un total de 16 situaciones). Se encuentra que los valores máximo y mínimo son los mismos en todos los casos (excepto para el caso cuando se prende $V10$) correspondiendo a:

$$Máximo = 758.22$$

$$Mínimo = 20.38$$

Para el caso con $V10$ prendida, el máximo es el mismo y el mínimo es de 17.71. A pesar de esto, en este caso se empleará también un valor mínimo de 20.38 para facilitar la comparación. A partir de estos valores, se determinan 9 escenarios distintos incrementando proporcionalmente el valor de la restricción de sumatoria de valores absolutos de los flujos desde el mínimo hasta el máximo. Para cada uno de estos 9 escenarios, se procede a hacer los cálculos que se detallan en la siguiente sección.

5.4.3.2. Descripción de las situaciones

Teniendo en cuenta las 7 fermentaciones presentes en el modelo núcleo compacto de E.coli (LAC, ETOH, AC, FOR, aKG, GLU, SUCC), se plantean los siguientes 16 casos:

- Sin fermentaciones y sin producción de biomasa
- Sin fermentaciones y con producción de biomasa
- Con una de las fermentaciones y sin producción de biomasa, para las 7 fermentaciones

- Con una de las fermentaciones y con producción de biomasa, para las 7 fermentaciones

Analizando cada uno de ellos para 9 valores distintos de la restricción de sumatoria de valores absolutos de los flujos planteada en la sección anterior más un caso sin dicha restricción, se tiene un total de 160 escenarios distintos. Para cada uno de estos 160 escenarios, se hacen los siguientes cálculos:

- Capacidad mínima y máxima de respiración (definida como $V22 + {}^{1}/_{2}\, V23$ ya que la entrada de oxígeno no está directamente representada en el modelo)
- Capacidad mínima y máxima de fermentación, si corresponde
- Capacidad mínima y máxima de producción de biomasa, si corresponde
- Capacidad mínima y máxima de ATP disponible ($V19$)

Asimismo, se utiliza el método de Monte Carlo directo para encontrar los siguientes valores:

- Volumen estimado del espacio de soluciones
- Distancia promedio al centro de masa del espacio de soluciones
- Distancia promedio al punto de máxima fermentación, si corresponde
- Distancia promedio al punto de máxima producción de biomasa, si corresponde

El procedimiento para los cálculos se detalla a continuación.

5.4.3.3. Procedimiento

Los cálculos de capacidad máxima y mínima para respiración, fermentación, producción de biomasa y ATP disponible se resuelven mediante un modelo en GLPK ajustado a cada situación, donde se prenden o apagan las reacciones correspondientes, se selecciona la función objetivo deseada, y se agrega la restricción de suma de valores absolutos de flujos acorde a la situación.

Se procede a hacer Monte Carlo directo en cada una de las situaciones con 10^7 iteraciones. Mediante este algoritmo, se obtiene el volumen del espacio de soluciones para cada situación, así como también la estimación del centro de masa.

Posteriormente, se aplica nuevamente el método de Monte Carlo a cada situación, también con 10^7 iteraciones, pero esta vez para calcular la distancia promedio al centro de masa de los puntos que caen adentro del espacio de soluciones. Se calcula también la distancia promedio al punto de máxima respiración, al punto de máximo crecimiento (si corresponde) y al punto de máxima fermentación (si corresponde).

Para encontrar los puntos de máxima respiración, máximo crecimiento y máxima fermentación, se utiliza un modelo en GLPK para cada situación. Cabe aclarar que estos modelos encuentran un punto máximo, que puede no ser único, existiendo la posibilidad de tener óptimos alternativos. Para obtener una solución única, una posibilidad sería asignar el valor de la función objetivo como una nueva restricción, y elegir otra función objetivo, repitiendo este proceso hasta llegar a una solución única. Para este trabajo no se lleva a cabo este procedimiento, pero se utiliza siempre el mismo modelo GLPK de forma de no ir cambiando el valor resultante en cada etapa del estudio.

5.4.3.4. Resultados

A continuación se grafican los resultados correspondientes a:

- Volumen del espacio de soluciones
- Volumen del espacio de soluciones elevado al inverso de la dimensión del modelo en cuestión
- Distancia euclídea promedio al centro de masa
- Volumen del espacio de soluciones elevado al inverso de la dimensión del modelo en cuestión dividido entre la distancia euclídea promedio al centro de masa

Todos estos resultados están en función del valor máximo asignado a la suma de los valores absolutos de los flujos.

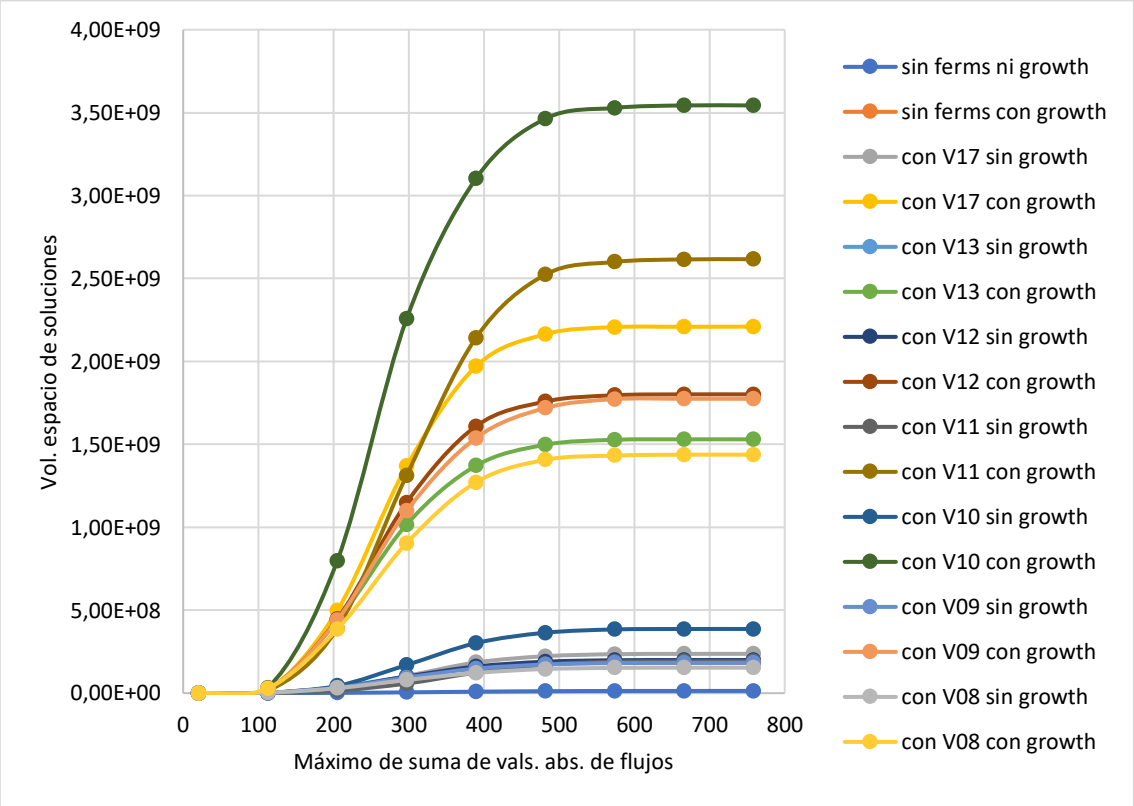


Ilustración 35 – Volumen del espacio de soluciones en función del máximo de la suma de valores absolutos de flujos

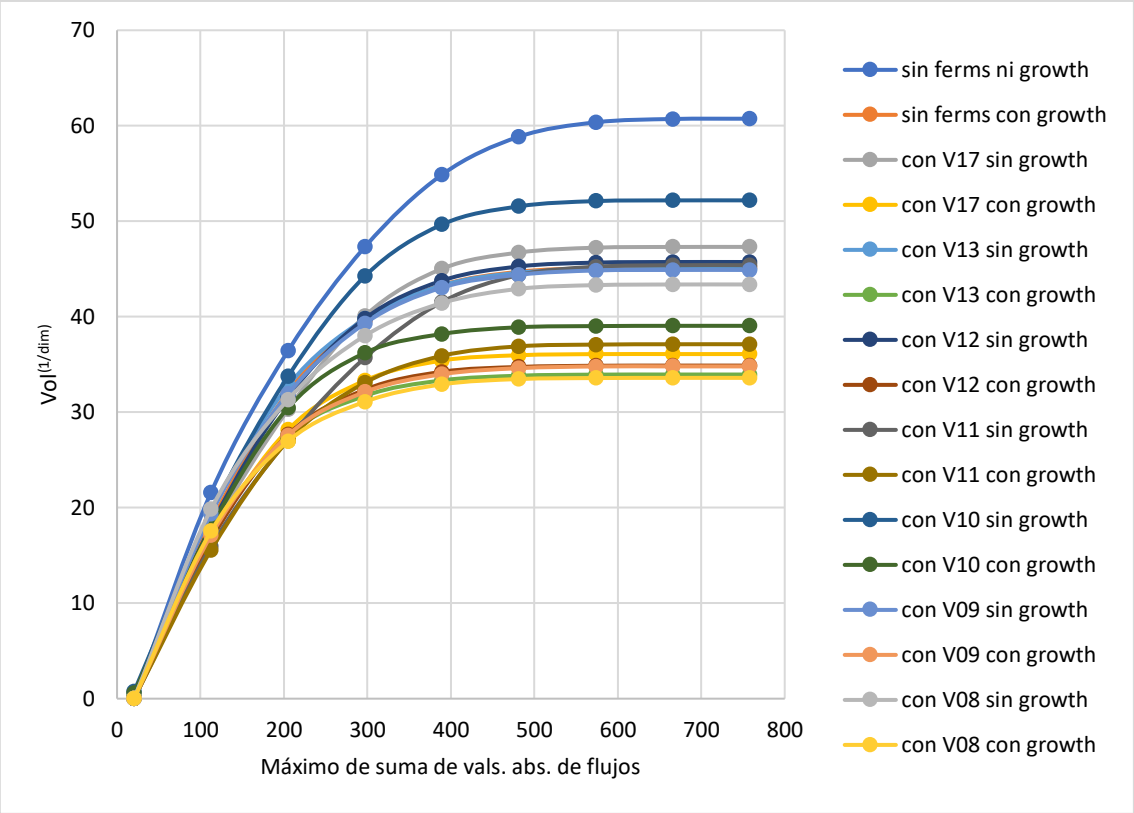


Ilustración 36 – Volumen del espacio de soluciones elevado a 1/dim

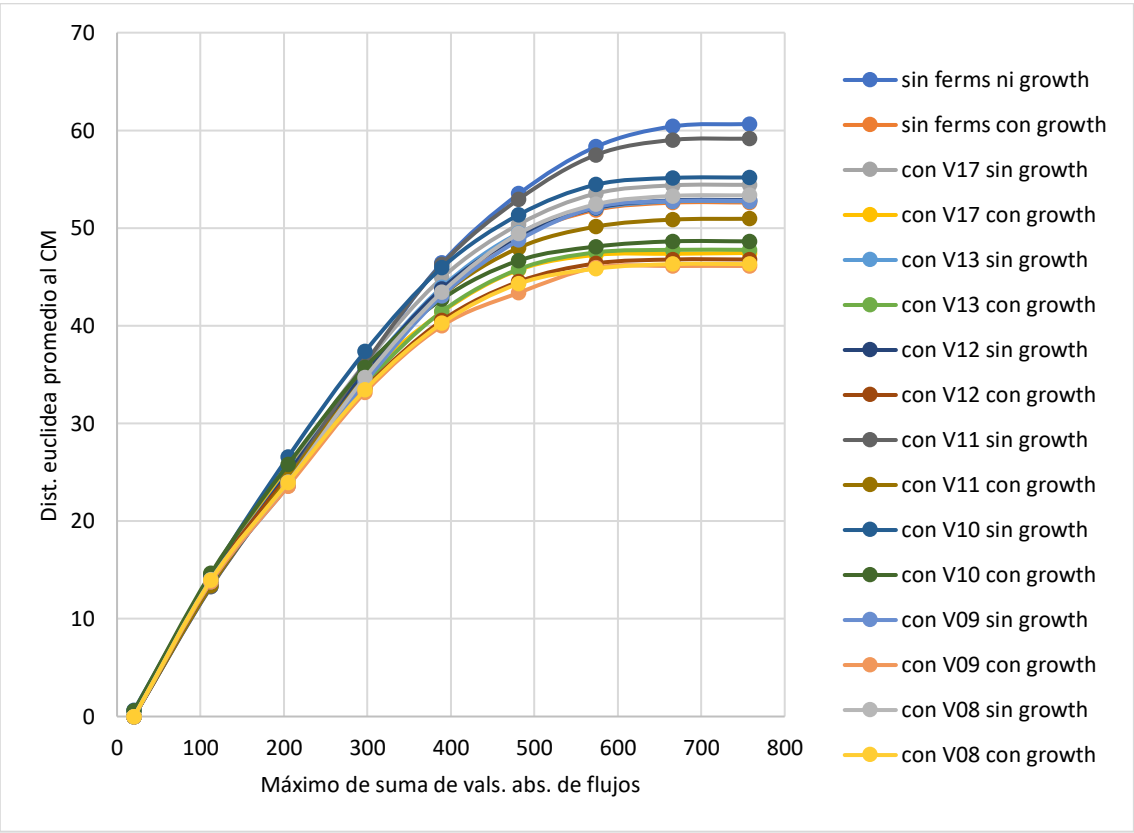


Ilustración 37 – Distancia promedio al centro de masa

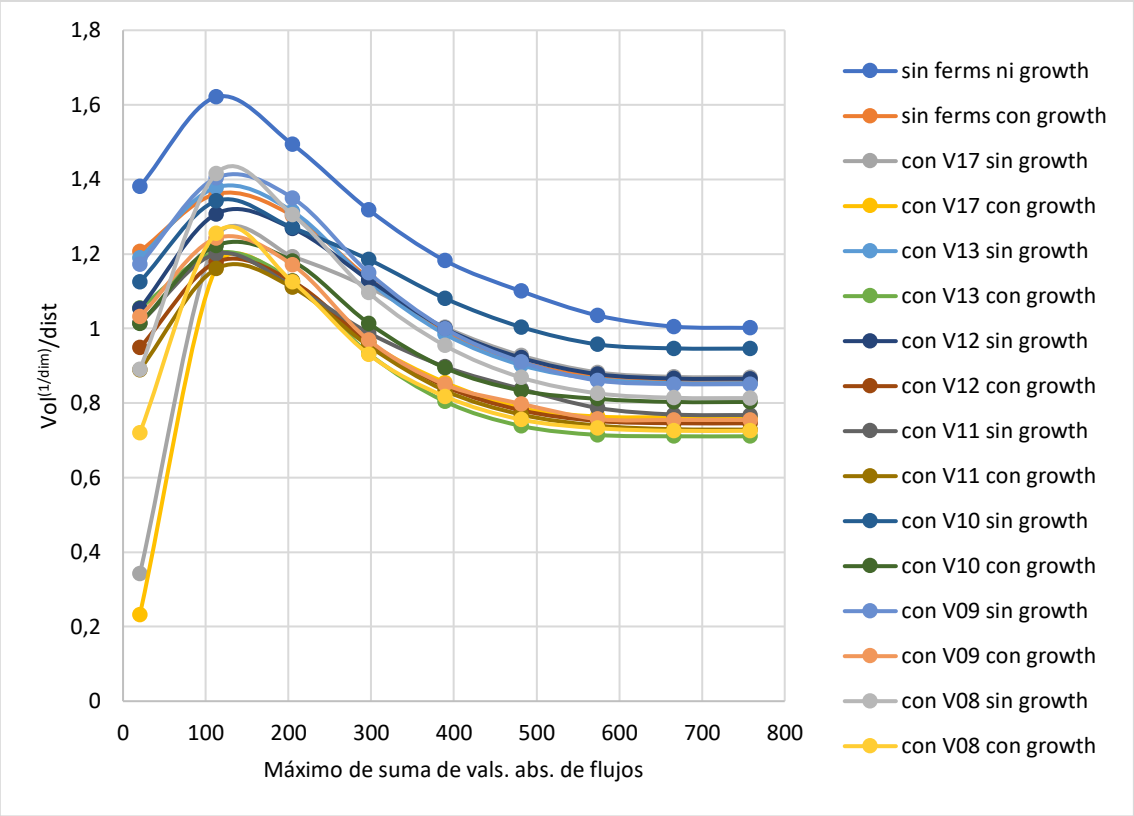


Ilustración 38 – Volumen del espacio de soluciones elevado a 1/dim dividido entre la distancia promedio al centro de masa

Se puede ver cómo la restricción de la suma de valores absolutos de flujos disminuye mucho el volumen del espacio de soluciones para valores muy bajos, pero no tiene un efecto significativo sobre el mismo para valores más cercanos al máximo. Esto se observa tanto en la Ilustración 35 como en la Ilustración 36. La distancia promedio al centro de masa también disminuye al restringirse la suma de valores absolutos de flujos. Al contrario de lo que ocurre para el volumen, ya para valores relativamente altos de la suma de valores absolutos de los flujos, la disminución de estos valores produce una disminución relativamente alta de la distancia promedio al centro de masa. Finalmente, la Ilustración 38 muestra la relación entre el volumen del espacio de soluciones elevado al inverso de la dimensión y la distancia promedio al centro de masa. Se observa un pico máximo para esta métrica con una restricción de suma de valores absolutos entre 100 y 200, que podría representar una zona de "mayor beneficio" para el microorganismo (donde un mayor espacio de soluciones implicaría un mayor repertorio de respuestas posibles y una menor distancia al centro de masa implicaría una mayor facilidad para acceder a todos los estados posibles del sistema).

5.4.4. Growth-volume trade-off

5.4.4.1. Introducción

En esta sección, se busca analizar la relación entre el volumen del espacio de soluciones y el crecimiento del microorganismo en el modelo núcleo compacto de E.coli. En primer lugar, se procede a calcular el máximo y mínimo flujo de VBM para el modelo en las distintas situaciones (sin fermentaciones y prendiéndolas de a una, 8 situaciones en total). Se obtienen siempre los mismos valores, que son:

$Máximo = 1.14$

$Mínimo = 0.00$

Posteriormente, se toma el intervalo entre el mínimo y el máximo y se eligen 6 puntos dentro del mismo (al 0%, al 20%, al 40%, al 60%, al 80% y al 100%). Estos puntos corresponden a distintos valores que se asignarán al LB de VBM. Es decir, para los cálculos correspondientes a este apartado, se va aumentando paulatinamente el LB de VBM en el modelo hasta llegar a tener el máximo flujo posible como LB, obligando al organismo a tener el máximo crecimiento posible.

Por un lado, se desea calcular la variabilidad del modelo (como se define en San Román et al. [8] según lo visto en la sección 2.3.1) en 8 situaciones (todas las fermentaciones apagadas y cada una de las 7 fermentaciones prendidas de a una) y para los distintos valores de LB de VBM (6 valores). Por otro lado, se desea calcular el volumen del espacio de soluciones del modelo en cada uno de estos casos (48 casos en total). El objetivo es comprobar si, para el modelo núcleo compacto de E.coli, se verifica el growth-flexibility trade-off, y estudiar la posibilidad de que también haya un growth-volume trade-off.

5.4.4.2. Procedimiento

5.4.4.2.1. Cálculo de variabilidades

La primera parte del trabajo consiste en calcular la variabilidad del modelo en las distintas situaciones y para los distintos valores de LB de VBM, para un total de 48 casos. Se procede a hacer FVA en los 48 casos, utilizando el software GLPK. El algoritmo de FVA resuelve:

```
for i in 1..25:
    maximize/minimize v[i]
    s.a:
      • balance{j in 1..13}: sum{i in 1..25} S[j,i]*v[i]=0
      • upperbound{i in 1..25}: v[i] <= Vmax[i]
      • lowerbound{i in 1..25}: v[i] >= Vmin[i]
```

donde se va cambiando la función objetivo para llegar a maximizar y minimizar todos los flujos del modelo. Para representar los 48 casos, se fija el valor de LB de VBM al valor correspondiente y se lleva a 0 el valor de UB de las fermentaciones que no estén presentes.

Una vez obtenidos los valores de máximo y mínimo de cada flujo en el modelo, se calcula el Δ para la situación:

$$\Delta = \frac{1}{r}\sum_{i \in R} \frac{v_i^{max} - v_i^{min}}{V_i^{max} - V_i^{min}}$$

donde r es la cantidad de reacciones internas, v_i^{max} y v_i^{min} son los valores de flujo máximo y flujo mínimo respectivamente para la reacción i bajo las nuevas condiciones, y V_i^{max} y V_i^{min} son los valores de flujo máximo y flujo mínimo respectivamente para la reacción i en las condiciones de referencia. En este modelo, r es 18 cuando no hay fermentaciones prendidas y 19 cuando hay una, y V_i^{max} y V_i^{min} son los valores

obtenidos de FVA para un LB de VBM de 0 con las fermentaciones del caso. Por lo tanto, siempre se tiene $\Delta = 1$ cuando el LB de VBM es 0.

Adicionalmente, se aplica el procedimiento para el caso con todas las fermentaciones prendidas. Esto no implica una dificultad adicional ya que se trata de un problema de programación lineal que se puede resolver fácilmente mediante GLPK. El volumen del espacio de soluciones de la situación con todas las fermentaciones prendidas, por el contrario, sí es un problema demasiado difícil de resolver mediante Monte Carlo directo, por lo que dicho volumen no se calcula para este trabajo.

5.4.4.2.2. Cálculo de volúmenes

La segunda parte del trabajo consiste en encontrar el volumen del espacio de soluciones para cada una de las 48 situaciones. Esto se hace mediante el método de Monte Carlo directo con 10^7 iteraciones. Es necesario generar una caja de muestreo para cada situación particular. Se calcula también el centro de masa del espacio de soluciones, la distancia al centro de masa, y la distancia al punto de máxima producción de biomasa, que se encuentra mediante GLPK. Este procedimiento es similar al usado en la sección 5.4.3. El código para Monte Carlo directo con cálculo del centro de masa está en el Anexo 12.

5.4.4.3. Resultados

La Tabla 46 y la Ilustración 39 presentan los resultados obtenidos para las variabilidades en función del porcentaje del máximo crecimiento como LB de VBM, en cada situación.

Tabla 46 – Variabilidades

	Δ								
% LB VBM	**sin ferms**	**con V08**	**con V09**	**con V10**	**con V11**	**con V12**	**con V13**	**con V17**	**con todas**
0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
20	0.827	0.848	0.845	0.850	0.839	0.853	0.853	0.858	0.873
40	0.654	0.667	0.665	0.699	0.679	0.706	0.707	0.715	0.735
60	0.482	0.487	0.485	0.549	0.518	0.543	0.522	0.558	0.581
80	0.297	0.295	0.294	0.353	0.347	0.323	0.313	0.334	0.367
100	0.073	0.066	0.066	0.066	0.070	0.066	0.066	0.062	0.042

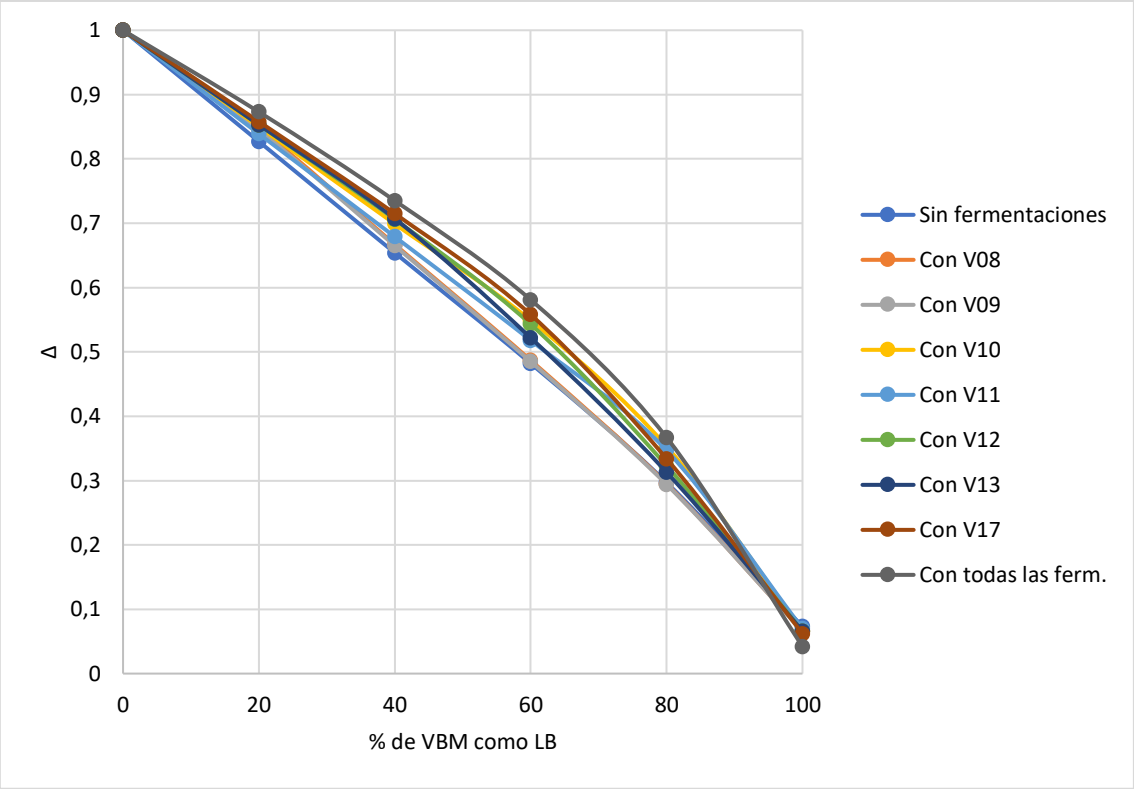


Ilustración 39 – Variabilidades

Se puede observar que la solución es aproximadamente lineal en todos los casos, y la variabilidad se acerca a 0 cuando se exige máximo crecimiento al modelo (es decir, cuando LB de VBM está al 100% del máximo crecimiento). Los resultados de San Román et al. [8] son similares a los obtenidos, según presentan los autores en los círculos negros de la Ilustración 40.

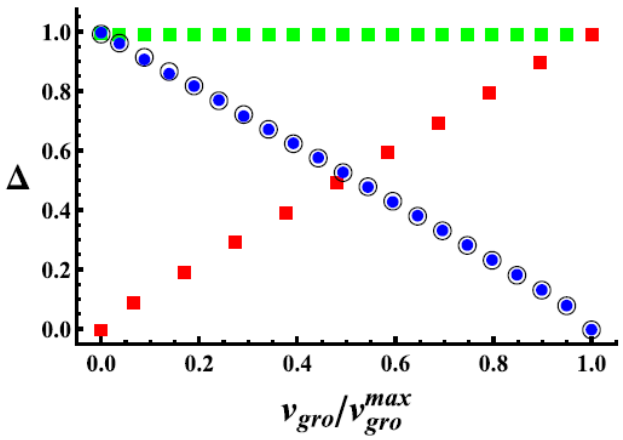


Ilustración 40 – "Total variability vs. growth rate, in aerobic conditions", tomado de [8]

En esta ilustración, los círculos negros corresponden a fijar el LB de crecimiento, mientras que se mantiene el UB de crecimiento en crecimiento máximo y la entrada de glucosa libre, entre 0 y 10. Esto muestra que una variabilidad alta exige que el crecimiento pueda ser regulado a valores mucho menores al máximo potencial.

En lo que respecta a los cálculos de volumen de las 48 situaciones, los resultados para volumen del espacio de soluciones, distancia promedio al centro de masa y distancia promedio al punto de máxima producción de biomasa se presentan en la Tabla 47, la Tabla 48 y la Tabla 49.

Tabla 47 – Resultados de volumen del espacio de soluciones

% LB VBM	sin ferms	con V08	con V09	con V10	con V11	con V12	con V13	con V17
0	1.87E+08	1.44E+09	1.77E+09	3.54E+09	2.62E+09	1.80E+09	1.53E+09	2.21E+09
20	9.87E+07	5.36E+08	6.50E+08	1.40E+09	1.34E+09	6.69E+08	5.80E+08	8.51E+08
40	3.90E+07	1.48E+08	1.88E+08	4.04E+08	4.95E+08	1.79E+08	1.59E+08	2.31E+08
60	9.36E+06	2.34E+07	2.71E+07	5.86E+07	1.01E+08	2.58E+07	2.33E+07	3.15E+07
80	6.71E+05	7.38E+05	9.18E+05	1.92E+06	4.44E+06	8.36E+05	7.52E+05	1.05E+06
100	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00

Tabla 48 – Resultados de distancia promedio al CM

% LB VBM	sin ferms	con V08	con V09	con V10	con V11	con V12	con V13	con V17
0	52.62	46.31	46.11	48.64	50.95	46.78	47.77	47.45
20	45.54	39.10	38.75	40.59	43.54	39.21	40.21	39.97
40	36.81	32.00	31.63	32.48	34.73	31.51	31.80	32.13
60	26.69	23.70	23.24	23.83	25.17	23.66	23.36	23.39
80	14.63	12.90	12.84	13.09	13.70	13.01	12.68	12.85
100	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Tabla 49 – Resultados de distancia promedio al pto. de máxima prod. de biomasa

% LB VBM	sin ferms	con V08	con V09	con V10	con V11	con V12	con V13	con V17
0	87.03	78.01	76.69	82.71	87.18	78.06	77.26	79.72
20	74.77	64.83	63.83	67.40	73.90	65.04	64.32	65.57
40	60.08	51.89	51.65	52.62	58.54	51.48	50.83	51.54
60	43.28	37.98	38.08	37.80	42.02	37.75	37.27	37.23
80	24.18	21.42	21.39	21.34	23.27	21.42	20.89	21.21
100	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Asimismo, se grafican los resultados de volumen y volumen elevado al inverso de la dimensión del espacio de soluciones, en función del porcentaje del máximo crecimiento como LB de VBM, para cada situación (Ilustración 41 e Ilustración 42, respectivamente). La dimensión del espacio de soluciones es de 6 para los casos con una fermentación y 5 para el caso sin fermentaciones. Se grafican también los resultados de distancia al centro de masa y distancia al punto de máxima fermentación para poder visualizarlos (Ilustración 43 e Ilustración 44, respectivamente).

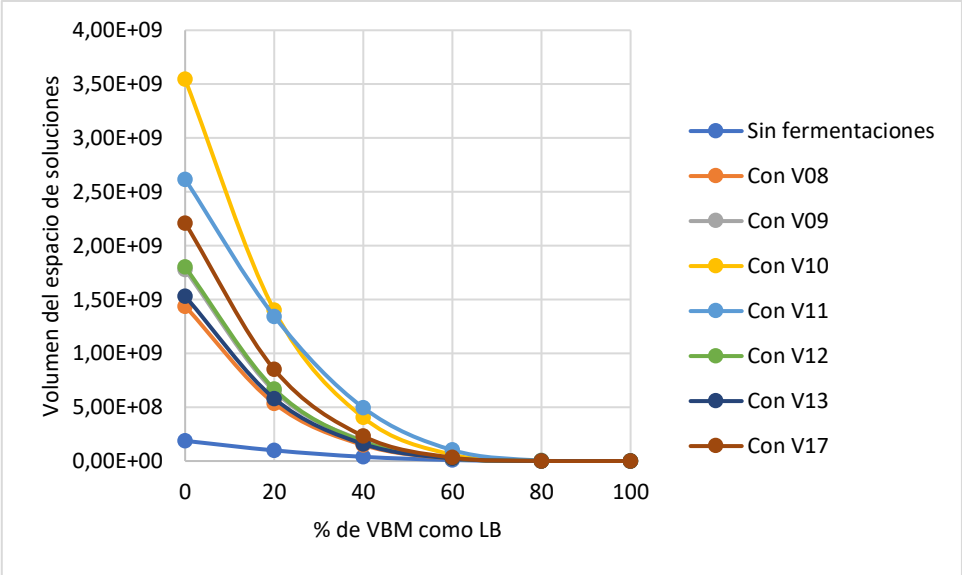


Ilustración 41 – Volumen del espacio de soluciones

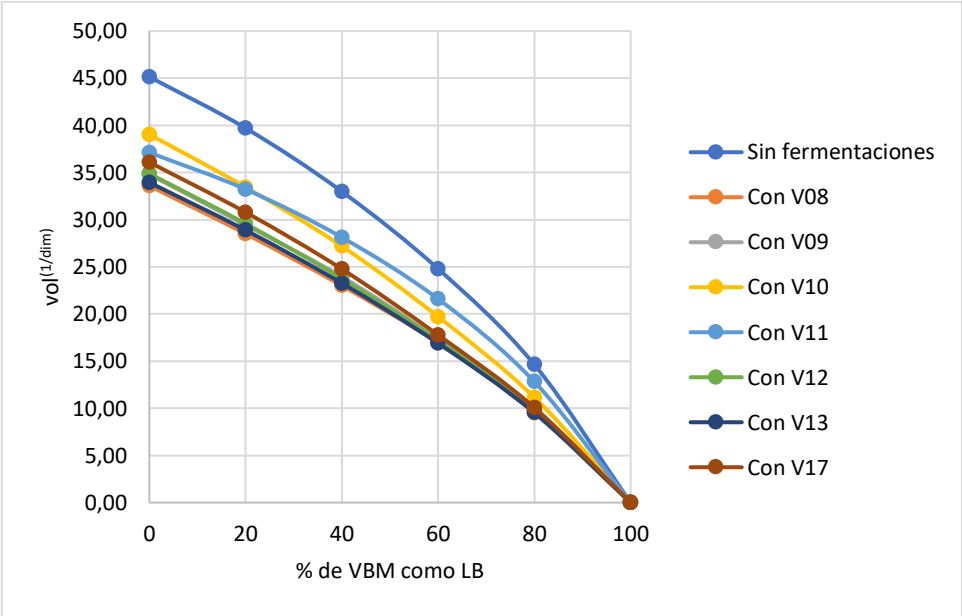


Ilustración 42 – Volumen elevado al inverso de la dimensión

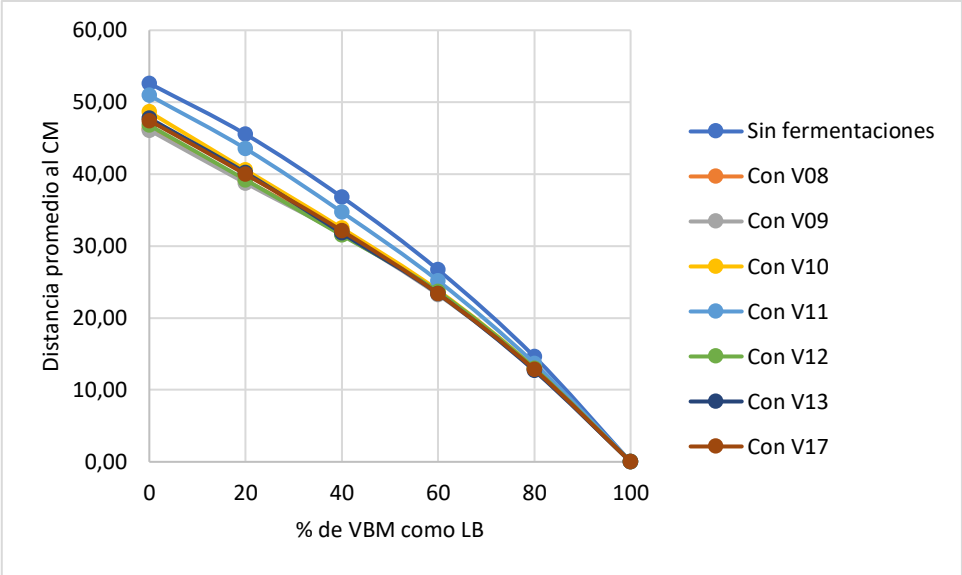


Ilustración 43 – Distancia promedio al CM

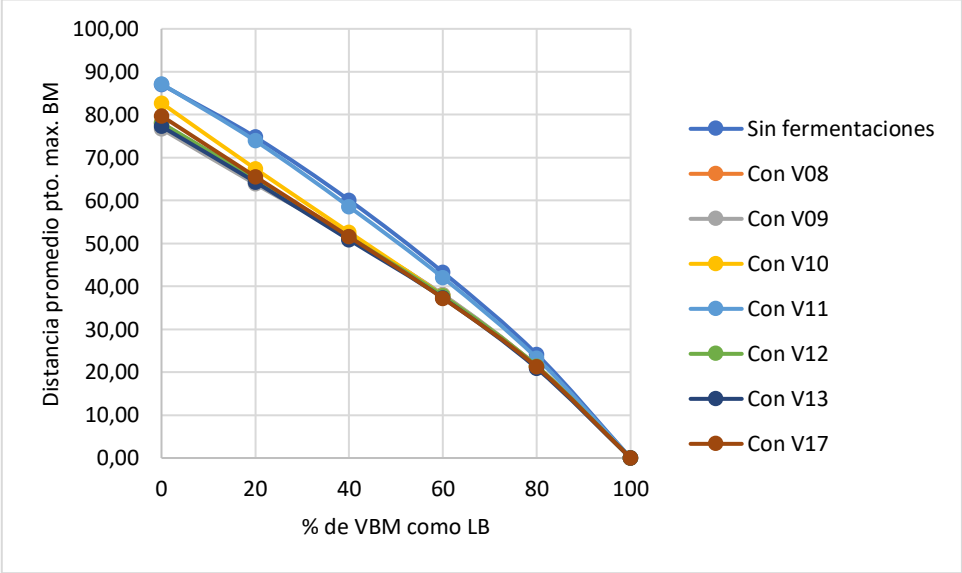


Ilustración 44 – Distancia promedio al pto. de máxima prod. de biomasa

Se observa cómo, al igual que para la variabilidad, el incrementar el LB de VBM también reduce el volumen del espacio de soluciones del modelo. Asimismo, se observa que se requiere reducir el LB de VBM a valores inferiores a alrededor de un 60% del crecimiento máximo para obtener un volumen que no sea despreciable con respecto al volumen máximo (obtenido con cota inferior igual a cero).

Finalmente, se representan los resultados de dividir el volumen elevado a uno sobre la dimensión por la distancia promedio al centro de masa y, por otra parte, por la distancia promedio al punto de máximo crecimiento (Ilustración 45 e Ilustración 46, respectivamente). En ambos casos se destaca el comportamiento diferencial (notoriamente ascendente) de las curvas correspondientes a la situación sin fermentaciones y a la situación con V11.

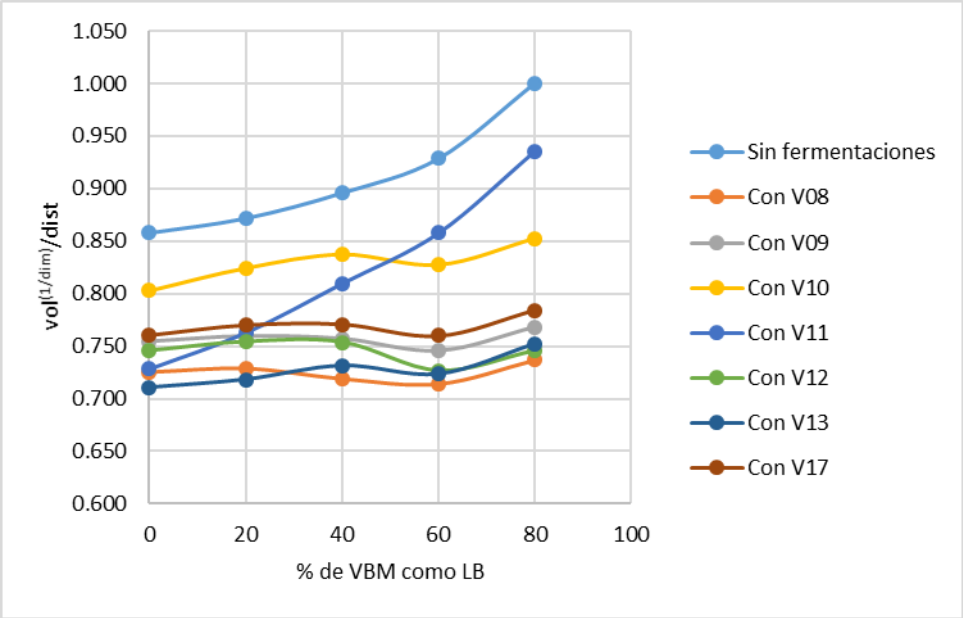


Ilustración 45 – Volumen elevado al inverso de la dimensión entre distancia promedio al CM

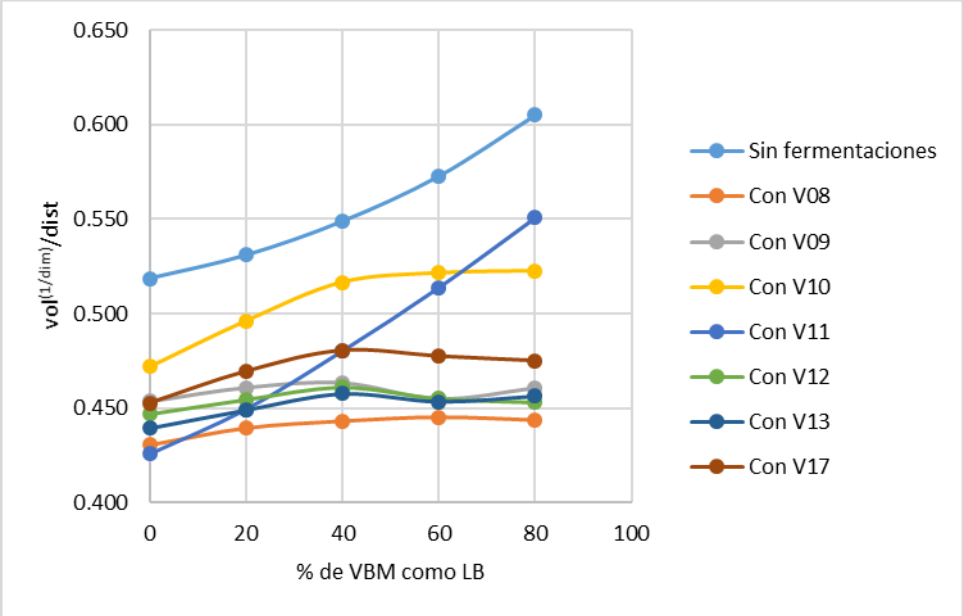


Ilustración 46 – Volumen elevado al inverso de la dimensión entre distancia promedio al pto. de máx. producción de biomasa

5.4.4.4. Con restricción de oxígeno

Se plantea una restricción en el consumo de oxígeno, haciendo que la entrada del mismo sea igual a 20. Como la entrada de oxígeno no está representada como un flujo en el modelo núcleo compacto de E.coli, se elige añadir la siguiente restricción:

$$V22 + \tfrac{1}{2}V23 = 20$$

Con esta restricción adicional, se procede a hacer el cálculo de volumen del espacio de soluciones en cada una de las 8 situaciones (sin fermentaciones y prendiendo una fermentación). Asimismo, se repiten los cálculos del growth-volume trade-off, teniendo

en cuenta que el máximo VBM posible para el modelo en cada situación va a ser distinto al presentado anteriormente debido a la restricción adicional.

Los resultados son similares a los obtenidos sin la restricción de oxígeno. En la Ilustración 47 se grafican los resultados de volumen en función del porcentaje del máximo crecimiento como LB de VBM, para cada situación.

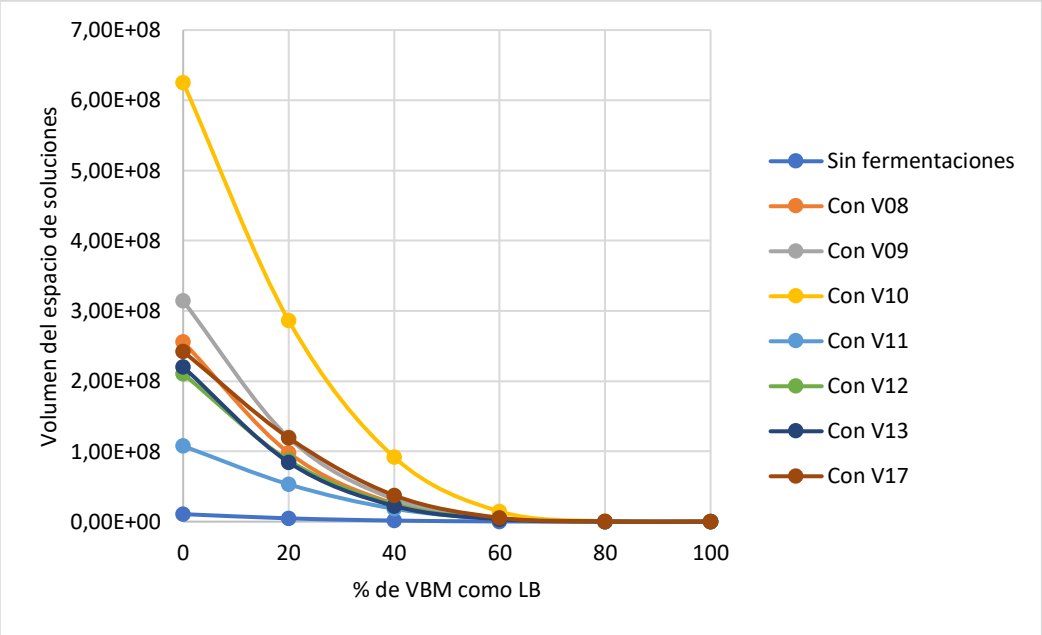


Ilustración 47 – Volumen del espacio de soluciones

5.4.4.5. Variando entrada de glucosa

Habiendo estudiado el comportamiento de los volúmenes de los espacios de soluciones al cambiar la restricción de suma de los valores absolutos de los flujos y el LB de VBM, ahora se busca determinar cómo cambian los volúmenes con la variación en el flujo de entrada de glucosa. Para simular esto, se decide variar el UB de V_{glc} ($V01$) a valores de 2, 4, 6, 8 y 10. Los resultados obtenidos se presentan en la Tabla 50.

Tabla 50 – Resultados de volumen del espacio de soluciones

UB GLC	sin ferms	con V08	con V09	con V10	con V11	con V12	con V13	con V17
2	3.71E+05	3.34E+05	4.16E+05	7.95E+05	1.18E+06	3.86E+05	3.32E+05	4.57E+05
4	8.25E+06	1.78E+07	2.13E+07	4.11E+07	5.22E+07	2.09E+07	1.81E+07	2.48E+07
6	3.93E+07	1.38E+08	1.66E+08	3.24E+08	3.58E+08	1.66E+08	1.44E+08	2.00E+08
8	1.04E+08	5.37E+08	6.55E+08	1.27E+09	1.21E+09	6.64E+08	5.65E+08	7.98E+08
10	1.87E+08	1.44E+09	1.77E+09	3.54E+09	2.62E+09	1.80E+09	1.53E+09	2.21E+09

Asimismo, la Ilustración 48 presenta el volumen estimado del espacio de soluciones en función de la cota superior del flujo de glucosa. En la Ilustración 49, se presenta el volumen estimado elevado al inverso de la dimensión, para cada situación. Se destaca que, en ambas ilustraciones, las curvas son crecientes.

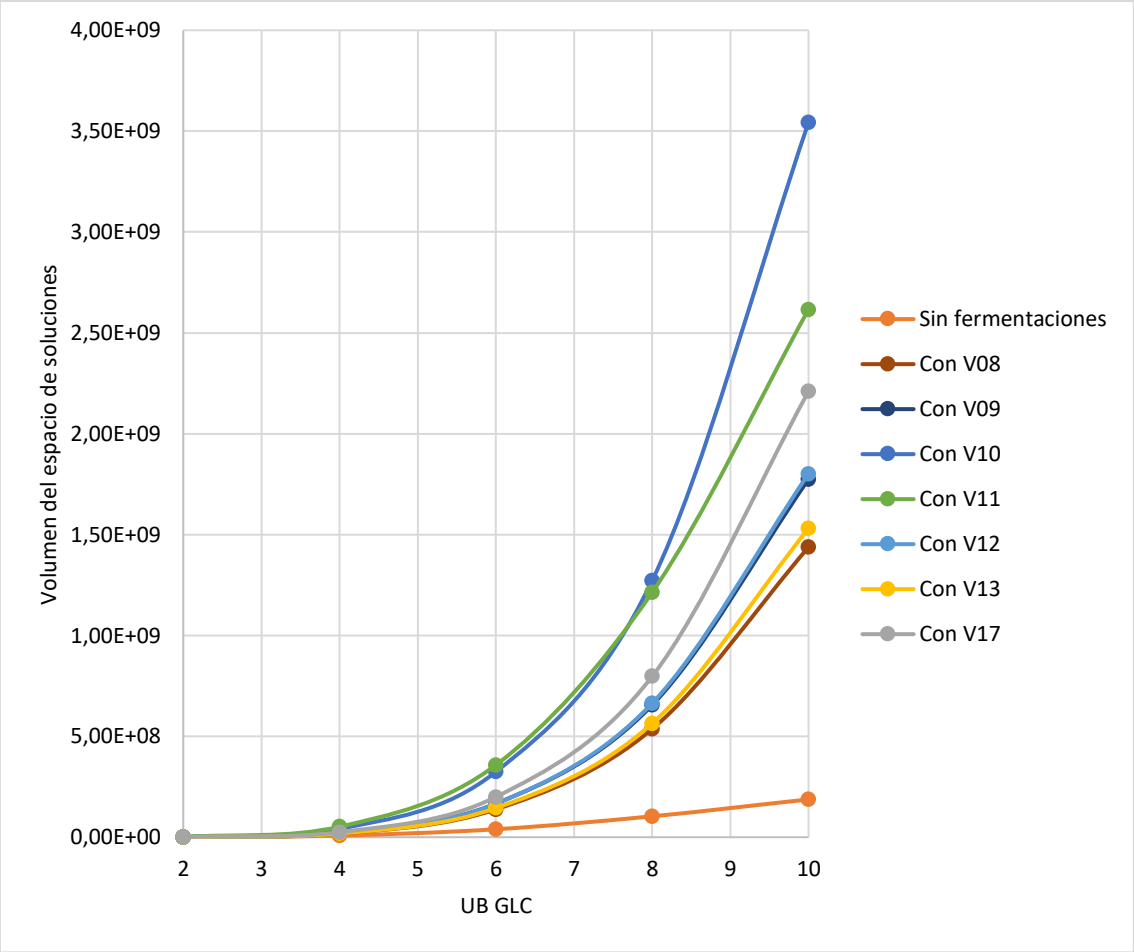


Ilustración 48 – Volumen en función de UB GLC

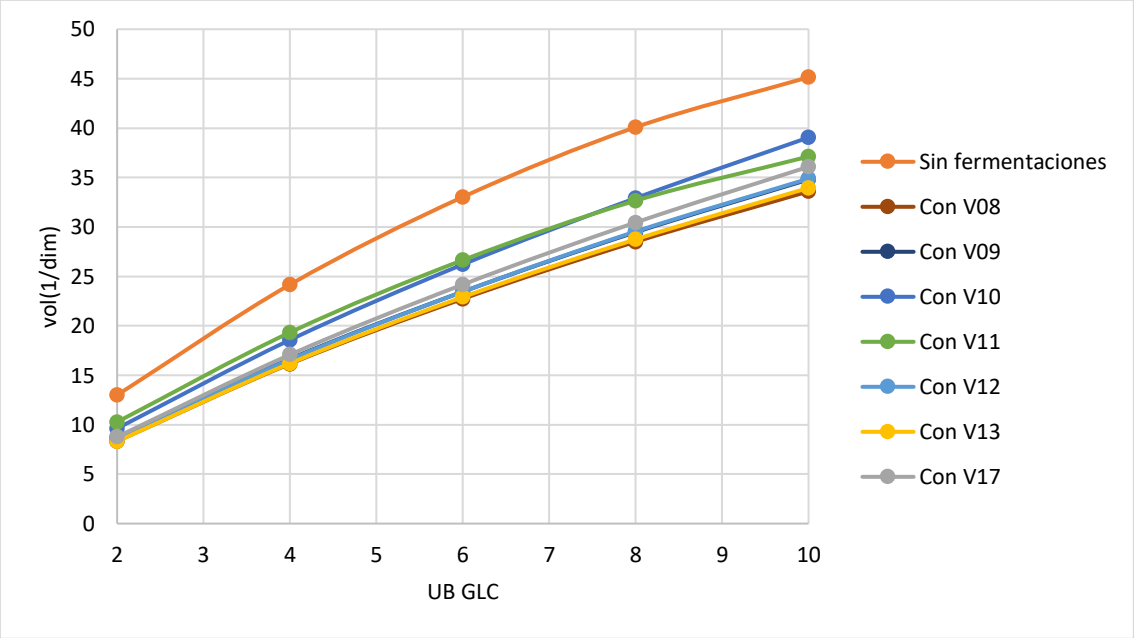


Ilustración 49 – Volumen elevado al inverso de la dimensión en función de UB GLC

Para la Ilustración 49, la relación entre las variables es aproximadamente lineal, comportándose en forma similar a cómo varía la variabilidad de flujo con la cota

superior del flujo de glucosa en San Román et al. [8], según presentan los autores en los círculos verdes de la Ilustración 50.

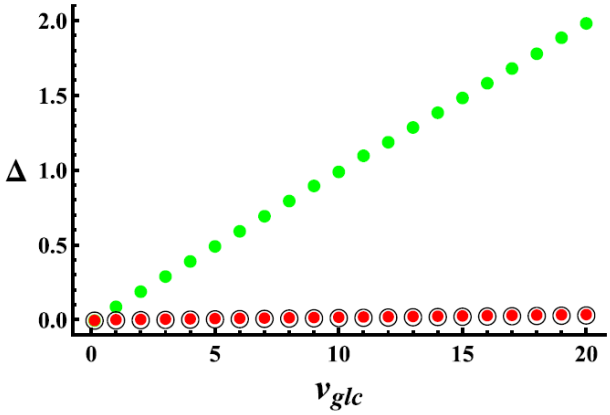


Ilustración 50 – "Components of flux variability vs. glucose uptake in aerobic conditions", tomado de [8]

6. Conclusiones

La primera parte de los resultados de este trabajo se centra en la aplicación de distintos métodos de cálculo de volumen al espacio de soluciones de un modelo que consiste en una ramificación simple, el caso de mayor interés y mayor riqueza de análisis en tres dimensiones (por lo tanto que es posible representar gráficamente de manera sencilla e intuitiva), siendo al mismo tiempo posible el cálculo analítico del volumen correspondiente. Este ejemplo se utilizó para ilustrar el concepto de espacio de soluciones en un modelo metabólico, la relación de su dimensionalidad con la cantidad de reacciones y metabolitos del sistema, y las dificultades que se presentan a la hora de calcular su volumen de forma eficiente.

En primer lugar, se empleó el método de Monte Carlo de forma directa, comparando dos maneras distintas de generar un espacio de referencia. Se llegó a la conclusión de que el método que utiliza la descomposición QR para generar una base ortonormal del espacio nulo de la matriz estequiométrica del modelo es un método efectivo y fácil de implementar, permitiendo generar espacios de referencia válidos para modelos metabólicos de tamaños mucho mayores al del modelo ramificado simple.

Posteriormente, se aplicaron al modelo ramificado simple dos formas alternativas de generar muestreos aleatorios para el cálculo de volumen: muestreo por importancia y Hit-and-Run. Para el caso de muestreo por importancia, se empleó una distribución triangular y una distribución exponencial truncada, viendo que ninguna de las dos aportaba mejoras significativas al muestreo en el caso de este modelo. Para Hit-and-Run, se puso a prueba el sampler del paquete COBRApy en conjunto con una metodología para su aplicación al cálculo de volúmenes. Luego de corregidos varios errores que se encontraron en el paquete, se concluyó que esta forma de calcular el volumen del espacio de soluciones permite aproximarse a su orden de magnitud, lo que podría ser de utilidad en modelos de mayores dimensiones.

Finalmente, se calculó satisfactoriamente el volumen del espacio de soluciones a partir de la integración de secciones de dicho volumen. Esto sirvió como una forma de validar el cálculo de secciones del volumen, lo que puede ser de utilidad en el análisis de un modelo si se desea estudiar el efecto de fijar flujos en un valor determinado. Asimismo, este análisis permitió entender mejor la relación que hay entre las distintas secciones en el espacio, al poder verlo de forma gráfica en el modelo ramificado simple.

La segunda parte de los resultados de este trabajo se centra en la construcción y validación del modelo núcleo compacto de E.coli, el cual se elaboró a partir del modelo núcleo de E.coli. A partir de distintas pruebas involucrando FBA y FVA, se comprobó que el modelo núcleo compacto de E.coli se comportaba de manera similar al modelo núcleo en los aspectos relevantes para este trabajo.

Posteriormente, se pasó a estudiar el espacio de soluciones del modelo núcleo compacto de E.coli mediante la aplicación de muestreo por importancia y mediante el uso del sampler de COBRApy. Siendo que este modelo resulta demasiado complejo para analizar mediante Monte Carlo directo, se decidió hacer simplificaciones sobre el mismo que consistieron mayormente en eliminar las distintas fermentaciones del

modelo (es decir, obligando al organismo a solo respirar), para poder así hacer una comparación de las distintas metodologías. Los resultados de la aplicación de muestreo por importancia, tanto para una distribución triangular como para una distribución exponencial truncada, no fueron satisfactorios, pero permitieron sacar conclusiones acera de la forma de los espacios de soluciones de modelos metabólicos y las dificultades que se presentan a la hora de intentar mejorar los muestreos obtenidos por Monte Carlo. Los resultados del uso del sampler de COBRApy, como se esperaba a partir de lo obtenido para el modelo ramificado simple, permitieron una aproximación al orden de magnitud del volumen del espacio de soluciones del modelo núcleo compacto de E.coli, lo que resulta prometedor como una posible herramienta para el análisis de modelos de grandes dimensiones.

En la parte final de los resultados de este trabajo, se hicieron distintos análisis sobre el modelo núcleo compacto de E.coli. Se estudió la existencia de sinergias entre las distintas fermentaciones a la hora de incrementar el volumen del espacio de soluciones del modelo, no encontrándose sinergias positivas entre ningún par de fermentaciones. Se estudió también el efecto de fijar las fermentaciones en un valor determinado en el volumen del espacio de soluciones, encontrándose que esto lleva a una reducción de dicho volumen en todos los casos. Interpretando el hecho de fijar una fermentación como tomar una sección de un espacio de soluciones en una dimensión más alta, se procedió a integrar los resultados para volver a obtener satisfactoriamente los volúmenes de los espacios de soluciones.

Por otro lado, se analizó el efecto de imponer una restricción a la suma de los valores absolutos de los flujos del modelo, observando su efecto en el volumen total del espacio de soluciones. Si bien se observó que la restricción disminuye el volumen del espacio de soluciones, esta no tiene un efecto considerable sobre el mismo para los valores más cercanos al máximo. Asimismo, se buscó también generar una métrica que relaciona el volumen del espacio de soluciones, su dimensionalidad, y la distancia promedio al centro de masa, generando un posible indicador del beneficio que un determinado estado del sistema puede tener para el microorganismo.

Finalmente, se comprobó la existencia del “growth-flexibility trade-off” y se estudió la existencia de un “growth-volume trade-off”, es decir, de una relación en la cual la imposición de una producción de biomasa mayor implica una reducción en el volumen del espacio de soluciones. Se observó que esta relación está presente en el modelo núcleo compacto de E.coli, lo que permite proponer al volumen del espacio de soluciones como una alternativa para caracterizar la flexibilidad que presenta el modelo bajo ciertas condiciones, para lo cual previamente se había hecho uso del parámetro de variabilidad calculado a partir de la resolución de problemas de programación lineal.

A modo de conclusión de este trabajo, puede decirse que los modelos generados (tanto el modelo ramificado simple como el modelo núcleo compacto de E.coli) se comportaron satisfactoriamente a la hora de poner a prueba las distintas metodologías e hipótesis planteadas. El uso de muestreo por importancia y Hit-and-Run para el cálculo de los volúmenes de los espacios de soluciones no resuelve el problema de la dificultad de calcular volúmenes en altas dimensiones, pero puede ser un camino para continuar explorando en futuros trabajos. Más allá de esto, en este trabajo queda evidenciado que

la aplicación del método de Monte Carlo para generar muestreos en un modelo relativamente sencillo permite llegar a conclusiones relevantes sobre el comportamiento del metabolismo de un determinado microorganismo.

7. Bibliografía

[1] P. Teixeira, H. Cancela, and L. Acerenza, "Studying metabolic metworks through Monte Carlo simulations," in *13th International Conference in Monte Carlo & Quasi-Monte Carlo Methods in Scientific Computing*, 2018.

[2] B. Ø. Palsson, *Systems Biology: Constraint-based Reconstruction and Analysis*. Cambridge University Press, 2015.

[3] F. Llaneras and J. Picó, "Stoichiometric modelling of cell metabolism," *J. Biosci. Bioeng.*, vol. 105, no. 1, pp. 1–11, 2008.

[4] J. D. Orth, I. Thiele, and B. Ø. Palsson, "What is flux balance analysis?," *Nat. Biotechnol.*, vol. 28, no. 3, pp. 245–248, 2010.

[5] J. D. Orth, B. Ø. Palsson, and R. M. T. Fleming, "Reconstruction and Use of Microbial Metabolic Networks: the Core Escherichia coli Metabolic Model as an Educational Guide," *EcoSal Plus*, vol. 4, no. 1, 2010.

[6] K. J. Kauffman, P. Prakash, and J. S. Edwards, "Advances in flux balance analysis," *Curr. Opin. Biotechnol.*, vol. 14, no. 5, pp. 491–496, 2003.

[7] N. E. Lewis, H. Nagarajan, and B. Ø. Palsson, "Constraining the metabolic genotype–phenotype relationship using a phylogeny of in silico methods," *Nat. Rev. Microbiol.*, vol. 10, pp. 291–305, 2012.

[8] M. San Román, H. Cancela, and L. Acerenza, "Source and regulation of flux variability in Escherichia coli," *BMC Syst. Biol.*, vol. 8, no. 1, pp. 1–11, 2014.

[9] R. Mahadevan and C. H. Schilling, "The effects of alternate optimal solutions in constraint-based genome-scale metabolic models," *Metab. Eng.*, vol. 5, no. 4, pp. 264–276, 2003.

[10] D. Segre, D. Vitkup, and G. M. Church, "Analysis of optimality in natural and perturbed metabolic networks," *Proc. Natl. Acad. Sci.*, vol. 99, no. 23, pp. 15112–15117, 2002.

[11] N. D. Price, J. Schellenberger, and B. Ø. Palsson, "Uniform sampling of steady-state flux spaces: Means to design experiments and to interpret enzymopathies," *Biophys. J.*, vol. 87, no. 4, pp. 2172–2186, 2004.

[12] T. Latzko, U. Jaekel, W. Wiechert, and K. Noh, "Markov Chain Monte Carlo Methods to Analyze the Steady-State Flux Solution Space of Metabolic Network Models," *Proc. 26th Eur. Simul. Model. Conf.*, no. Mcmc, pp. 1–7, 2012.

[13] N. Chaudhary, K. Tøndel, R. Bhatnagar, V. A. P. Martins Dos Santos, and J. Puchałka, "Characterizing the optimal flux space of genome-scale metabolic reconstructions through modified latin-hypercube sampling," *Mol. Biosyst.*, vol. 12, no. 3, pp. 994–1005, 2016.

[14] S. J. Wiback, I. Famili, H. J. Greenberg, and B. Palsson, "Monte Carlo sampling can be used to determine the size and shape of the steady-state flux space," *J. Theor. Biol.*, vol. 228, no. 4, pp. 437–447, 2004.

[15] U. Systems Biology Research Group, "E Coli Core," 2013. [Online]. Available:

http://systemsbiology.ucsd.edu/Downloads/EcoliCore. [Accessed: 15-Jul-2019].

[16] Free Software Foundation Inc., “GLPK (GNU Linear Programming Kit),” 2012. [Online]. Available: https://www.gnu.org/software/glpk. [Accessed: 15-Jul-2019].

[17] A. Ebrahim, B. Ø. Palsson, J. A. Lerman, and D. R. Hyduke, “COBRApy: COnstraints-Based Reconstruction and Analysis for Python,” *BMC Syst Biol.*, vol. 7, no. 74, p. doi: 10.1186/1752-0509-7-74, 2013.

[18] R. Heinrich and S. Schuster, *The Regulation of Cellular Systems*. Chapman & Hall, 1996.

[19] J. Schellenberger, “Monte Carlo Simulation in Systems Biology,” University of California, San Diego, 2010.

[20] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, 2000.

[21] A. B. Owen, “Monte Carlo theory, methods and examples,” 2013. [Online]. Available: https://statweb.stanford.edu/~owen/mc/. [Accessed: 15-Jul-2019].

[22] E. C. Anderson, “Monte Carlo Methods and Importance Sampling,” 1999. [Online]. Available: http://ib.berkeley.edu/labs/slatkin/eriq/classes/guest_lect/mc_lecture_notes.pdf. [Accessed: 15-Jul-2019].

[23] W. Megchelenbrink, M. Huynen, and E. Marchiori, “optGpSampler: An improved tool for uniformly sampling the solution-space of genome-scale metabolic networks,” *PLoS One*, vol. 9, no. 2, 2014.

[24] S. Chawla, “Metropolis method, volume estimation,” 2007. [Online]. Available: http://pages.cs.wisc.edu/~shuchi/courses/880-S07/scribe-notes/lecture26-1.pdf. [Accessed: 15-Jul-2019].

[25] B. K. S. Chung and D. Y. Lee, “Flux-sum analysis: A metabolite-centric approach for understanding the metabolic network,” *BMC Syst. Biol.*, vol. 3, pp. 1–10, 2009.

Anexos

Anexo 1: Determinación de rangos de muestreo mediante GLPK para modelo ramificado simple

```
#Planteo del modelo flux split en GLPK
param W{j in 1..3, i in 1..2};
param Vmax{j in 1..3};
param Vmin{j in 1..3};
var x{i in 1..2};
minimize z: x[2];
s.t. multmax{j in 1..3}: sum{i in 1..2} W[j,i]*x[i] <= Vmax[j];
s.t. multmin{j in 1..3}: sum{i in 1..2} W[j,i]*x[i] >= Vmin[j];

data;
param W : 1 2 :=
1 -0.5774 0.5774
2 0.7887 0.2113
3 0.2113 0.7887 ;
param Vmax := 1  6
              2  8
              3  10;
param Vmin := 1  0
              2  0
              3  0;

end;
```

Anexo 2: Monte Carlo directo con descomposición QR en modelo ramificado simple

```
#Ejemplo de Price et al. (Figure 1)
#Monte Carlo con cálculo según descomp. QR

import random
import time
import numpy
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
from matplotlib import cm

#Defino la matriz a partir de la descomp. QR de P'
W=[[-0.5774,0.5774],[0.7887,0.2113],[0.2113,0.7887]]
W=numpy.array(W)
seed=123
random.seed(seed)
n=10000000

xin=[]
yin=[]
zin=[]
xout=[]
yout=[]
zout=[]
maximos=[6,8,10]
minimos=[0,0,0]
est_vol=0
start_time=time.time()

for j in range (0,n):
    randvec=[]
    randvec.append(random.uniform(-2.1957,8))
    randvec.append(random.uniform(0,12.1957))
    randvec=numpy.array(randvec)
    flujoscurrent=numpy.dot(W,randvec)
    if all (flujoscurrent[k]<=maximos[k] for k in range (0,3)):
        if all (flujoscurrent[k]>=minimos[k] for k in range (0,3)):
            xin.append(flujoscurrent[0])
            yin.append(flujoscurrent[1])
            zin.append(flujoscurrent[2])
            est_vol=est_vol+1
        else:
            xout.append(flujoscurrent[0])
            yout.append(flujoscurrent[1])
            zout.append(flujoscurrent[2])
    else:
        xout.append(flujoscurrent[0])
        yout.append(flujoscurrent[1])
        zout.append(flujoscurrent[2])

end_time=time.time()

vol_muestreo=(8+2.1957)*(12.1957+0)
est_desv=(vol_muestreo**2)*(est_vol/n)*(1-(est_vol/n))/(n-1)
desviacion=est_desv**0.5
```

```
print('Cantidad de hits =',est_vol)
print('Hit fraction =',est_vol/n)
print('Volumen del espacio de muestreo =',vol_muestreo)
print('Volumen del espacio de soluciones =',vol_muestreo*est_vol/n)
print('Desviacion estandar =',desviacion)
print('Elapsed time =',end_time-start_time)

fig=plt.figure(1)
ax1=fig.add_subplot(111,projection='3d')

ax1.scatter(xin,yin,zin,c='deeppink',edgecolor='none',label='in')
ax1.scatter(xout,yout,zout,c='dimgray',edgecolor='none',label='out')
ax1.set_xlabel('v1')
ax1.set_xlim(0,10)
ax1.set_ylabel('v2')
ax1.set_ylim(0,10)
ax1.set_zlabel('v3')
ax1.set_zlim(0,10)
ax1.view_init(elev=33,azim=-120)
plt.legend()
plt.show()
```

Anexo 3: Monte Carlo directo con paralelepípedos en modelo ramificado simple

```
def mc(n,seed):
    import random
    import time
    import numpy

    xin=[]
    yin=[]
    zin=[]
    xout=[]
    yout=[]
    zout=[]

    random.seed(seed)

    flujoscurrent=[]
    for i in range (0,3):
        flujoscurrent.append(0)

    #B=[[0,6],[8,0],[8,6]]
    #B=[[0,6],[10,-6],[10,0]]
    B=[[10,-8],[0,8],[10,0]]
    maximos=[6,8,10]
    minimos=[0,0,0]

    est_vol=0
    start_time=time.time()

    for j in range (0,n):
        esc1=random.random()
        esc2=random.random()
        flujoscurrent[0]=B[0][0]*esc1+B[0][1]*esc2
        flujoscurrent[1]=B[1][0]*esc1+B[1][1]*esc2
        flujoscurrent[2]=B[2][0]*esc1+B[2][1]*esc2
        if all (flujoscurrent[k]<=maximos[k] for k in range (0,3)):
            if all (flujoscurrent[k]>=minimos[k] for k in range (0,3)):
                est_vol=est_vol+1

    end_time=time.time()

    B=numpy.array(B)
    Bt=numpy.matrix.transpose(B)
    vol_paral=(numpy.linalg.det(numpy.dot(Bt,B)))**0.5

    est_desv=(vol_paral**2)*(est_vol/n)*(1-(est_vol/n))/(n-1)
    desviacion=est_desv**0.5


    print('Cantidad de hits =',est_vol)
    print('Hit fraction =',est_vol/n)
    print('Volumen del paralelepipedo =',vol_paral)
    print('Volumen del espacio de soluciones =',vol_paral*est_vol/n)
    print('Desviacion estandar =',desviacion)
    print('Elapsed time =',end_time-start_time)
```

Anexo 4: Cálculo de volumen con COBRApy en modelo ramificado simple

```
#Ejemplo de Price et al. (Figure 1)
#Muestreos sucesivos dentro de caja

import time
import numpy as np
import cobra
from cobra import Model, Reaction, Metabolite
from cobra.flux_analysis import sample

model=Model('flux_split_en_caja')

reaction1 = Reaction('V1')
reaction2 = Reaction('V2')
reaction3 = Reaction('V3')
reaction4 = Reaction('x1')
reaction5 = Reaction('x2')

reaction1.lower_bound = -1000
reaction2.lower_bound = -1000
reaction3.lower_bound = -1000
reaction4.lower_bound = -2.1957
reaction5.lower_bound = 0

reaction1.upper_bound = 1000
reaction2.upper_bound = 1000
reaction3.upper_bound = 1000
reaction4.upper_bound = 8
reaction5.upper_bound = 12.1957

A=Metabolite('A')
alfa=Metabolite('alfa')
beta=Metabolite('beta')
gama=Metabolite('gama')

reaction1.add_metabolites({A:1,alfa:-1})
reaction2.add_metabolites({A:1,beta:-1})
reaction3.add_metabolites({A:-1,gama:-1})
reaction4.add_metabolites({alfa:-0.5774,beta:0.7887,gama:0.2113})
reaction5.add_metabolites({alfa:0.5774,beta:0.2113,gama:0.7887})

model.add_reactions([reaction1])
model.add_reactions([reaction2])
model.add_reactions([reaction3])
model.add_reactions([reaction4])
model.add_reactions([reaction5])

#Cantidad de muestras en cada iteracion
n=10000

#Constraints adicionales
co1=model.problem.Constraint(model.reactions.V1.flux_expression,ub=6)
co2=model.problem.Constraint(model.reactions.V1.flux_expression,lb=0)
co3=model.problem.Constraint(model.reactions.V2.flux_expression,ub=8)
co4=model.problem.Constraint(model.reactions.V2.flux_expression,lb=0)
co5=model.problem.Constraint(model.reactions.V3.flux_expression,ub=10)
co6=model.problem.Constraint(model.reactions.V3.flux_expression,lb=0)
```

```
#Sin constraints
start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 0 =",end_time-start_time)
np.savetxt("muestras0_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego primer constraint para muestreo
model.add_cons_vars([co1])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 1 =",end_time-start_time)
np.savetxt("muestras1_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego segundo constraint para muestreo
model.add_cons_vars([co2])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 2 =",end_time-start_time)
np.savetxt("muestras2_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego tercer constraint para muestreo
model.add_cons_vars([co3])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 3 =",end_time-start_time)
np.savetxt("muestras3_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego cuarto constraint para muestreo
model.add_cons_vars([co4])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 4 =",end_time-start_time)
np.savetxt("muestras4_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego quinto constraint para muestreo
model.add_cons_vars([co5])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 5 =",end_time-start_time)
np.savetxt("muestras5_cobrapy.csv",b,delimiter=";",newline="\n")

#Agrego sexto constraint para muestreo
```

```
model.add_cons_vars([co6])

start_time=time.time()
s=sample(model,n)
b=np.asmatrix(s)
end_time=time.time()
print("elapsed time 6 =",end_time-start_time)
np.savetxt("muestras6_cobrapy.csv",b,delimiter=";",newline="\n")
```

Anexo 5: Intercambios relativos a paquete COBRApy

Pablo Teixeira <pablo.txm@gmail.com> 25 de febrero de 2018, 14:38
Para: cobra pie <cobra-pie@googlegroups.com>
Hello,

I'm trying to use COBRApy to get samples of steady-state fluxes for a metabolic model. I've been using the most simple model I could come up with to try to understand how this works, so I've decided to reproduce the flux split found on figure 1A of this article: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1304643/ (I'm attaching it here as well).

The code I'm using is as follows:

```
from cobra import Model, Reaction, Metabolite
model=Model('flux_split')

reaction1 = Reaction('V1')
reaction2 = Reaction('V2')
reaction3 = Reaction('V3')

reaction1.lower_bound = 0
reaction2.lower_bound = 0
reaction3.lower_bound = 0

reaction1.upper_bound = 6
reaction2.upper_bound = 8
reaction3.upper_bound = 10

A=Metabolite('A')

reaction1.add_metabolites({A:-1})
reaction2.add_metabolites({A:-1})
reaction3.add_metabolites({A:1})

model.add_reactions([reaction1])
model.add_reactions([reaction2])
model.add_reactions([reaction3])
```

Then, I've tried using the sampling commands in COBRApy, however I don't get the results I expected (many of the sample points I get are not in the null space of my stoichiometric matrix, and plotting the results I don't see the solution space that I'm supposed to get). I've done:

```
from cobra.flux_analysis import sample
s = sample(model, 100)
```

And I've also tried doing:

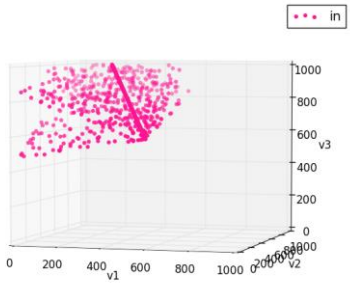
```
from cobra.flux_analysis.sampling import OptGPSampler, ACHRSampler
optgp = OptGPSampler(model, processes=3, thinning=1000)
optgp_samples = optgp.sample(1000)
optgp_samples = pd.DataFrame(optgp_samples, columns=[r.id for r in
model.reactions])

import numpy as np
b=np.asmatrix(optgp_samples)
```

I'm attaching the plot I get from this last one here, so you can see the samples I get don't make sense (they should be evenly distributed and in only one plane, and also they don't match the solution space described in the paper).

I'll appreciate any help you can give me. I assume the way I'm building my model is probably wrong, but I've tried to do this in the simplest way possible and I just can't see what's wrong.

Thank you,
Pablo



Christian Diener <ch.diener@gmail.com> 26 de febrero de 2018, 18:11
Para: cobra pie <cobra-pie@googlegroups.com>
Hi Pablo,

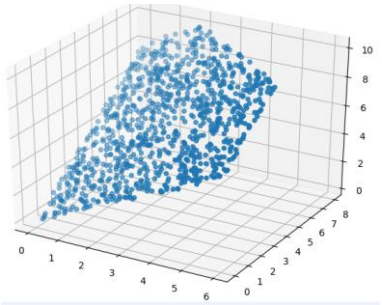
how you construct the model is completely correct. That particular model is pretty difficult for artificial centering since it gets stuck easily and due to the lower dimension there is a chance of choosing a center which makes the sampler walk exclusively along a line.

I will try to find a fix and add the model to the tests.

Cheers
Christian

Christian Diener <ch.diener@gmail.com> 26 de febrero de 2018, 18:26
Para: cobra pie <cobra-pie@googlegroups.com>
Okay, just to let you know I found the issue. This actually only occurs if the nullspace dimnsion of your problem is 2 or less which is not too common. I will add a fix. In the appendix the sample with the fix, which looks pretty much like the figure you includes (just rotated).

Cheers
Christian



Pablo Teixeira <pablo.txm@gmail.com> 26 de febrero de 2018, 19:08
Para: cobra pie <cobra-pie@googlegroups.com>

Hi Christian,

Thank you so much! I think your plot looks exactly like what it's supposed to. Would you mind letting me know how I can fix it?

Thanks again,
Pablo

Christian Diener <ch.diener@gmail.com> 27 de febrero de 2018, 17:51
Para: cobra pie <cobra-pie@googlegroups.com>
Hi Pablo, I am completing my fix and will send a PR today. So it should be fixed in a new release soon. The major problem was that there were only two unique sampling directions in the problem setup and thus the mean of those would fall on a line between the two mich made sampling only walk along that line. I fixed that now by checking for that condition. The model actually turned out to have several other little numerical instabilities so it helped me to fix a variety of issues in the sampling code, so thanks for providing this model :)

Note that even with all of the mentioned changes the sampler still generates infeasible samples from time to time (|S*v| > 1e-6). Those are pretty close to feasible, but you can exclude them anyways by doing something like:

```Python
s = optgp.sample(100)
valid = optgp.validate(s) == "v"
s_valid = s[valid]
```

Alternatively you can set the `nproj` parameter to 1 when creating the `OptGPSampler` object. This will check feasibility on every iteration but will also make sampling pretty slow. Thus, I recommend the above solution since for your particular model the infeasible samples are pretty close to feasibility and only occur in less than 5% of all samples...

Cheers
Christian

Pablo Teixeira <pablo.txm@gmail.com> 28 de febrero de 2018, 00:20
Para: cobra pie <cobra-pie@googlegroups.com>
Awesome, thank you Christian!

Pablo Teixeira <pablo.txm@gmail.com> 1 de marzo de 2018, 16:39
Para: cobra pie <cobra-pie@googlegroups.com>
Hello Christian,

Sorry to bother you again. I tried using a bigger toy model, this time with 12 reactions and 7 metabolites. For one of the fluxes in particular, the samples I get are always fixed to the maximum value allowed, which should not be the case. I can email you my model if necessary. Is this a known issue with sampling in COBRApy?

Thanks a lot!

Best,
Pablo

Christian Diener <ch.diener@gmail.com> 1 de marzo de 2018, 22:21
Para: cobra pie <cobra-pie@googlegroups.com>
Yes, send it to me and I will take a look.

Thanks
Christian

Christian Diener <ch.diener@gmail.com> 2 de marzo de 2018, 15:31
Para: cobra pie <cobra-pie@googlegroups.com>
Ok thanks. Again a problem with the sampler :(But it should be fixed now in https://github.com/opencobra/cobrapy/pull/672. So basically there was a problem if you sampled a completely irreversible model since it would never include any minimizations in the search directions.

Cheers
Christian

Christian Diener <ch.diener@gmail.com> 6 de marzo de 2018, 14:28
Para: cobra pie <cobra-pie@googlegroups.com>
Ok, the fixes have been merged but it will probably take a little longer for a new release. You can test it installing the development version of cobrapy (for instance in a virtual environment) using:

```bash
pip install git+https://github.com/opencobra/cobrapy@devel
```

Cheers
Christian

Anexo 6: Importance sampling con distribución triangular en modelo ramificado simple

```
#Modelo flux split
#Monte Carlo con calculo segun descomposicion QR
#IMPORTANCE SAMPLING - DISTRIBUCION TRIANGULAR

import random
import time
import numpy
import scipy
from scipy import linalg

#Defino la matriz a partir de la descomp. QR de P'
W=[[-0.5774,0.5774],[0.7887,0.2113],[0.2113,0.7887]]
W=numpy.array(W)

#Importar flujos maximos y minimos
maximos=[6,8,10]
minimos=[0,0,0]

#Monte Carlo (IMPORTANCE SAMPLING con distr TRIANGULAR)

cotas_sup=[8,12.1957]
cotas_inf=[-2.1957,0]
punto_medio=[1.90215,7.09785]

suma_cuadrados=0

random.seed(123)
n=10000000
contar_adentro=0
est_vol=0
start_time=time.time()

for j in range (0,n):
    xrand=[]
    coefs=[]
    for i in range (0,2):

varx=numpy.random.triangular(cotas_inf[i],punto_medio[i],cotas_sup[i])
        fdistorig=1/(cotas_sup[i]-cotas_inf[i])
        if varx<=punto_medio[i]:
            fdisttriang=2*(varx-cotas_inf[i])/((cotas_sup[i]-
cotas_inf[i])*(punto_medio[i]-cotas_inf[i]))
        else:
            fdisttriang=2*(cotas_sup[i]-varx)/((cotas_sup[i]-
cotas_inf[i])*(cotas_sup[i]-punto_medio[i]))
        Lx=fdistorig/fdisttriang
        xrand.append(varx)
        coefs.append(Lx)
    flujoscurrent=numpy.dot(W,xrand)
    if all (flujoscurrent[k]<=maximos[k] for k in range (0,3)):
        if all (flujoscurrent[k]>=minimos[k] for k in range (0,3)):
            contar_adentro=contar_adentro+1
            est_vol=est_vol+numpy.prod(coefs)
            suma_cuadrados=suma_cuadrados+numpy.prod(coefs)**2
```

```
end_time=time.time()

print('Elapsed time =',end_time-start_time)
print('Cantidad de iteraciones =',n)
print('Cantidad de puntos adentro =',contar_adentro)
print('Estimador =',est_vol/n)
print('Est desviacion estandar del estimador=',((suma_cuadrados/(n-1)-
n*((est_vol/n)**2)/(n-1))/n)**0.5)

vol_caja=numpy.prod(numpy.array(cotas_sup)-numpy.array(cotas_inf))
print('Volumen de la caja =',vol_caja)
print('Volumen estimado del espacio de soluciones =',vol_caja*est_vol/n)
print('Est desviacion estandar del volumen =',(((suma_cuadrados/(n-1)-
n*((est_vol/n)**2)/(n-1))/n)**0.5)*vol_caja)
```

Anexo 7: Importance sampling con distribución exponencial truncada en modelo ramificado simple

```
#Modelo flux split
#Monte Carlo con calculo segun descomposicion QR
#IMPORTANCE SAMPLING - EXPONENCIAL TRUNCADA

import random
import time
import numpy
import scipy
from scipy import linalg

#Defino la matriz a partir de la descomp. QR de P'
W=[[-0.5774,0.5774],[0.7887,0.2113],[0.2113,0.7887]]
W=numpy.array(W)

#Importar flujos maximos y minimos
maximos=[6,8,10]
minimos=[0,0,0]

#Monte Carlo (IMPORTANCE SAMPLING con distr EXPONENCIAL TRUNCADA)

cotas_sup=[8,12.1957]
cotas_inf=[-2.1957,0]
punto_medio=[1.90215,7.09785]

random.seed(123)
n=10000000
lam=2
contar_adentro=0
est_vol=0
suma_cuadrados=0
start_time=time.time()

for j in range (0,n):
    xrand=[]
    coefs=[]
    for i in range (0,2):
        U=random.random()
        if random.choice(["der","izq"])=="der":
            auxvarx=-(1/lam)*numpy.log(1-U*(1-numpy.exp(-lam*(cotas_sup[i]-
punto_medio[i]))))
            varx=punto_medio[i]+auxvarx
            fdistorig=1/(cotas_sup[i]-cotas_inf[i])
            fdistexp=0.5*(lam*numpy.exp(-lam*auxvarx)/(1-numpy.exp(-
lam*(cotas_sup[i]-punto_medio[i]))))
            Lx=fdistorig/fdistexp
        else:
            auxvarx=-(1/lam)*numpy.log(1-U*(1-numpy.exp(-lam*(punto_medio[i]-
cotas_inf[i]))))
            varx=punto_medio[i]-auxvarx
            fdistorig=1/(cotas_sup[i]-cotas_inf[i])
            fdistexp=0.5*(lam*numpy.exp(-lam*auxvarx)/(1-numpy.exp(-
lam*(punto_medio[i]-cotas_inf[i]))))
            Lx=fdistorig/fdistexp
        xrand.append(varx)
        coefs.append(Lx)
```

```
    flujoscurrent=numpy.dot(W,xrand)
    if all (flujoscurrent[k]<=maximos[k] for k in range (0,3)):
        if all (flujoscurrent[k]>=minimos[k] for k in range (0,3)):
            contar_adentro=contar_adentro+1
            est_vol=est_vol+numpy.prod(coefs)
            suma_cuadrados=suma_cuadrados+numpy.prod(coefs)**2

end_time=time.time()

print('Elapsed time =',end_time-start_time)
print('Cantidad de iteraciones =',n)
print('Cantidad de puntos adentro =',contar_adentro)
print('Estimador =',est_vol/n)
print('Est  desviacion  estandar  del  estimador=',((suma_cuadrados/(n-1)-
n*((est_vol/n)**2)/(n-1))/n)**0.5)

vol_caja=numpy.prod(numpy.array(cotas_sup)-numpy.array(cotas_inf))
print('Volumen de la caja =',vol_caja)
print('Volumen estimado del espacio de soluciones =',vol_caja*est_vol/n)
print('Est  desviacion  estandar  del  volumen  =',(((suma_cuadrados/(n-1)-
n*((est_vol/n)**2)/(n-1))/n)**0.5)*vol_caja)
```

Anexo 8: Construcción del modelo compact core

Módulo I: glc_D[e] + 2 NAD + 2 pi + ADP -> pyr + pep + 2 NADH + 3 h + ATP + 2 H2O {9r}

GLCpts (D-Glucose transport via PEP:Pyr PTS): glc_D[e] + pep -> g6p + pyr

PGI (Glucose-6-phosphate isomerase): g6p = f 6p

PFK (Phosphofructokinase): ATP + f6p -> ADP + fdp + h

FBA (Fructose-bisphosphate aldolase): fdp = dhap + g3p

TPI (Triose-phosphate isomerase): dhap = g3p

GAPD (Glyceraldehyde-3-phosphate dehydrogenase): g3p + NAD + pi = 13 dpg + h + NADH

PGK (Phosphoglycerate kinase): 13 dpg + ADP = 3pg + ATP

PGM (Phosphoglycerate mutase): 3pg = 2pg

ENO (Enolase): 2pg = pep + H2O

Módulo II: pep + ADP + h -> pyr + ATP {1r}

PYK (Pyruvate kinase): pep + ADP + h -> pyr + ATP

Módulo III: pyr + coa + NAD -> accoa + CO2 + NADH {1r}

PDH (Pyruvate dehydrogenase): pyr + coa + NAD -> accoa + CO2 + NADH

Módulo IV: accoa + oaa + NADP + H2O -> akg + CO2 + NADPH + h + coa {3r}

CS (Citrate synthase): accoa + oaa + H2O -> cit + coa + h

ACONT (Aconitase): cit = icit

ICDHyr (Isocitrate dehydrogenase NADP): icit + NADP = akg + CO2 + NADPH

Módulo V: akg + nh4[e] + NADPH + h = glu + NADP + H2O {2r}

GLUDy (Glutamate dehydrogenase NADP): akg + nh4 + NADPH + h = glu + NADP + H2O

NH4t (ammonia reversible transport): nh4[e] <=> nh4

Módulo VI: glu + nh4[e] + ATP -> gln + ADP + h + pi {1r}*

GLNS (Glutamine synthetase): glu + nh4[e] + ATP -> gln + ADP + h + pi

NH4t (ammonia reversible transport): nh4[e] <=> nh4

(*) Una reacción ya fue usada en el Módulo V.

Módulo VII: akg + gln + h + NADPH -> 2 glu-L + NADP {1r}

GLUSy (Glutamate synthase NADPH): akg + gln + h + NADPH -> 2 glu-L + NADP

Módulo VIII: pyr + NADH + 2 h = lac[e] + NAD + h[e] {2r}

LDH_D (D-Lactate dehydrogenase): pyr + NADH + h = lac + NAD

D_LACt2 (D-Lactate transport via proton symport): h + lac = h[e] + lac[e]

Módulo IX: accoa + 3 h + 2 NADH = etoh[e] + coa + 2 NAD + h[e] {3r}

ACALD (Acetaldehyde dehydrogenase - acetylating): accoa + h + NADH = acald + coa + NAD

ALCD2x (Alcohol dehydrogenase - ethanol): acald + h + NADH = etoh + NAD
ETOHt2r (Ethanol reversible transport via proton symport): etoh + h = etoh[e] + h[e]

Módulo X: accoa + ADP + pi + h = ac[e] + coa + ATP + h[e] {3r}
PTAr (Phosphotransacetylase): accoa + pi = actp + coa
ACKr (Acetate kinase): actp + ADP = ac + ATP
ACt2r (Acetate reversible transport via proton symport): ac + h = ac[e] + h[e]

Módulo XI: coa + pyr -> accoa + for[e] {2r}
PFL (Pyruvate formate lyase): coa + pyr -> accoa + for
FORti (Formate transport via diffusion): for = for[e]

Módulo XII: akg + h = akg[e] + h[e] {1r}
AKGt2r (2-Oxoglutarate reversible transport via symport): akg + h = akg[e] + h[e]

Módulo XIII: glu + h = glu[e] + h[e] {1r}
GLUt2r (L-glutamate transport via proton symport – reversible): glu + h = glu[e] + h[e]

Módulo XIV: akg + NAD + ADP + pi -> succ + co2 + NADH + ATP {2r}
AKGDH (2-Oxoglutarate dehydrogenase): akg + coa + NAD -> co2 + NADH + succoa
SUCOAS (Succinyl-CoA synthetase - ADP-forming): ADP + pi + succoa = ATP + coa + succ

Módulo XV: q8 + succ -> fum + q8h2 {1r}
SUCDi (Succinate dehydrogenase - irreversible): q8 + succ -> fum + q8h2

Módulo XVI: fum + NAD + h2o = oaa + NADH + h {2r}
FUM (Fumarase): fum + h2o = mal-L
MDH (Malate dehydrogenase): mal-L + NAD = h + NADH + oaa

Módulo XVII: h[e] + succ -> h + succ[e] {1r}
SUCCt3 (Succinate transport out via proton antiport): h[e] + succ -> h + succ[e]

Módulo XVIII: co2 + h2o + pep -> h + oaa + pi {1r}
PPC (Phosphoenolpyruvate carboxylase): co2 + h2o + pep -> h + oaa + pi

Módulo XIX: ATP + h20 -> ADP + pi + h {1r}
ATPM (ATP maintenance requirement): ATP + h20 -> ADP + pi + h

Módulo XX: NAD + NADPH -> NADH + NADP {1r}

NADTRHD(NAD transhydrogenase): NAD + NADPH -> NADH + NADP

Módulo XXI: 2 h[e] + NADH + NADP -> 2 h + NAD + NADPH {1r}

THD2 (NAD(P) transhydrogenase): 2 h[e] + NADH + NADP -> 2 h + NAD + NADPH

Módulo XXII: 2 q8h2 + o2 + ADP + pi + h -> 2 q8 + ATP + 3 h2o {2r}

CYTBD (Cytochrome oxidase bd - ubiquinol-8:2 prot): 2 h + 1/2 o2 + q8h2 -> h2o + 2 h[e] + q8

ATPS4r (ATP synthase - four protons for one ATP): ADP + 4 h[e] + pi = ATP + h2o + 3 h

Módulo XXIII: NADH + 1/2 o2 + 5/4 ADP + 5/4 pi + 9/4 h -> NAD + 5/4 ATP + 9/4 h2o {1r}*

NADH16 (NADH dehydrogen - ubiquinone-8 & 3 prot): 4 h + NADH + q8 -> 3 h[e] + NAD + q8h2

CYTBD (Cytochrome oxidase bd - ubiquinol-8:2 prot): 2 h + 1/2 o2 + q8h2 -> h2o + 2 h[e] + q8

ATPS4r (ATP synthase - four protons for one ATP): ADP + 4 h[e] + pi = ATP + h2o + 3 h

(*) Dos reacciones ya fueron usadas en el Módulo XXII.

Módulo XXIV: fum + q8h2 -> q8 + succ **{1r}**

FRD7 (Fumarate reductase) fum + q8h2 -> q8 + succ

Módulo XXV: 4 h + NADH + q8 -> 3 h[e] + NAD + q8h2 {0r}*

NADH16 (NADH dehydrogen - ubiquinone-8 & 3 prot): 4 h + NADH + q8 -> 3 h[e] + NAD + q8h2

(*) la reacción ya fue usada en el Módulo XXIII.

Módulo XXVI: BM reaction

(3.7478) accoa + (59.8100) ATP + (0.2557) gln-L + (4.9414) glu-L+ (59.8100) h2o + (3.5470) NAD + (13.0279) NADPH + (1.7867) oaa + (0.5191) pep + (2.8328) pyr -> (59.8100) ADP + (4.1182) akg + (3.7478) coa + (59.8100) h + (3.5470) NADH + (13.0279) NADP + (59.8100) pi

Anexo 9: Modelo E.coli core en formato GLPK y COBRApy

Modelo core en formato GLPK

```
param S{j in 1..72, i in 1..95};
param Vmax{i in 1..95};
param Vmin{i in 1..95};
var v{i in 1..95};

maximize z: v[13];

#maximize z: sum{i in 1..95} v[i];

s.t. balance{j in 1..72}: sum{i in 1..95} S[j,i]*v[i]=0;
s.t. upperbound{i in 1..95}: v[i] <= Vmax[i];
s.t. lowerbound{i in 1..95}: v[i] >= Vmin[i];

data;

param S : 1  2      3      4      5      6      7      8      9      10     11
       12    13     14     15     16     17     18     19     20     21     22
       23    24     25     26     27     28     29     30     31     32     33
       34    35     36     37     38     39     40     41     42     43     44
       45    46     47     48     49     50     51     52     53     54     55
       56    57     58     59     60     61     62     63     64     65     66
       67    68     69     70     71     72     73     74     75     76     77
       78    79     80     81     82     83     84     85     86     87     88
       89    90     91     92     93     94     95:=
1      0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      1      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      1      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0
2      0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      -1     0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      -1
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0
3      0     0      0      0      0      0      0      0      0      0      0
       0     -1.496 0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      -1     0      1
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0
4      0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
       0     0      0      0      0      0      0      0      0      0      0
```

```
        0       0       0       0       0       0       0       0       0       0       0
        0       -1      0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       1       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
5       0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       -1      0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
6       0       0       -1      0       0       1       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
7       0       0       0       0       0       -1      0       0       0       0       0
        0       0       0       0       0       0       0       0       -1      0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
8       -1      1       0       0       0       0       0       0       0       1       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
9       0       -1      0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       -1      0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
10      1       0       0       0       0       0       0       0       0       0       0
        0       -3.7478         0       -1      0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       -1      0       0       0
        0       0       0       0       0       1       0       1       0       0       0
        0       0       0       0       0       -1      0       0       0       0       0
        0       0       0       0       0       0       0       0
```

```
11    0     0     0     1     -1    0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
12    0     0     1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0     0     0     0     0     0
      0     0     0     0     0     0     0
13    0     0     1     0     0     0     2     0     0     0     1
      -1    59.81 0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     1     1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     1     0     0     1     0     0
      0     0     1     0     0     -1    0     0     0     0     0
      0     1     0     0     0     0     0
14    0     0     0     0     0     0     0     -1    1     0     0
      0     4.1182 0    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     1     0     -1
      0     0     0     1     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
15    0     0     0     0     0     0     0     0     -1    0     0
      0     0     0     0     0     0     0     0     0     0     -1
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
16    0     0     0     0     0     0     -1    0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     1     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
17    0     0     -1    0     0     0     -1    0     0     0     -1
      1     -59.81 0    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     -1    -1    0     0     0
```

```
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0      -1       0       0      -1       0       0
        0       0      -1      -1       0       1       0       0       0       0       0
        0      -1       0       0       0       0       0
18      0       0       0      -1       0       0       0       0       0       0       0
        0       0       0       1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
19      0       0       0       0       0       0       0       1       0       0       0
        0       0       1       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       1       0       1       0       0       0       0       0       1       1
        0       0       0       0       1       0       0       0       0       0       0
        0      -1       1       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
20      0       0       0       0       0       0       0       0       0       0       0
        0       0      -1       0       0       0       0       0       0       0       0
       -1       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
21     -1       0       0       0       0       0       0      -1       0       0       0
        0   3.7478      0       1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       1       0       0       0       0
        0       0       0       0      -1       0      -1       0       0       0       0
        0       0       0       0       1       0       0       0       0       0       0
        0      -1       0       0       0       0       0
22      0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       1       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0      -1
23      0       0       0       0       0       0       0       0       0       0       0
        0  -0.3610      0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       1       0       0      -1       0
```

```
24    0     0     0     0     0     0     0     0     0    -1     0
      0     0     0     0     0     0     0     1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
25    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0     0     0
      0    -1     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
26    0     0     0     0     0     0     0     0     0     0     0
      0    -0.0709      0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     1     0     0
      0     1     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0    -1     0     1     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     1     0     0     1     0
27    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0    -1    -1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     1     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
28    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     1    -1     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     1     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
29    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0    -1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0    -1     1     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
30    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0    -1     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
     -1     0     0     0     0     0     0     0     0     0     0
```

```
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
31    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0    -1
      0    -1     1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      1     0     0     0     0     0     0
32    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0    -1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0    -1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
33    0     0     0     0     0     0     0     0     0     0     0
      0  -0.129   0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     1     0     0     0     0
      0     0     0     0    -1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0    -1     0     1     1     1
34    0     0     0     0     0     0     0     0     0     0     0
      0  -0.205   0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0    -1     0     1     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
35    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0    -1     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0    -1     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
36    0     0     0     0     0     0     0     0     0     0     0
      0  -0.2557        0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     1     1     0    -1
     -1     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0
```

```
37    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0    -1     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
38    0     0     0     0     0     0     0     0     0     0     0
      0    -4.9414     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0    -1     1
      2     1     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
39    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
     -1     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
40    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0    -1     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
41    0     0     0     1    -1     0     0     0     0     0    -1
      1    -59.81 0    -1     1     0     1     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     0     0     0
      0    -1     0     0     0     0     0    -1    -1    -1     0
      0     0     1     0     0     0    -1     0     0     0     0
      0     0     0     0     0     0     0     0     0    -1     0
      0    -1     0    -1     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
42    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0    -1     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0    -1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
43    1     0     0     0     0     1     0     0     1     1     1
      3    59.81  0     1    -2     1     0     1     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     1     0     0
      0     0     2     1     1     0     1     1     1     0    -1
```

	1	0	0	0	0	1	1	2	1	0	0
	-4	0	0	0	0	1	0	0	0	1	0
	1	1	0	2	0	-1	1	0	0	2	1
	0	0	0	2	0	0	0				
44	0	0	0	0	0	-1	0	0	-1	0	0
	-4	0	0	0	2	-1	0	-1	0	0	0
	0	0	0	0	0	0	0	0	-1	0	0
	0	0	0	0	0	0	0	0	-1	0	0
	0	0	-2	0	0	0	0	0	0	0	0
	-1	0	0	0	0	0	0	-2	0	0	0
	3	0	0	0	0	0	0	0	0	0	0
	-1	0	0	0	0	0	-1	0	0	-2	-1
	0	0	0	-2	0	0	0				
45	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	-1	-1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0				
46	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	-1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0				
47	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	-1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	-1
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0				
48	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	1	-1	-1	-1
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0				
49	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	-1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	-1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0				

```
50    -1    0     0     0     0     0     0     -1    0     -1    0
      0     -3.547 0    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     -1    0     0     0     0     0     0
      0     0     0     0     0     -1    0     0     -1    -1    0
      1     -1    0     0     -1    0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     1     0     0     0
51    1     0     0     0     0     0     0     1     0     1     0
      0     3.547 0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0     0     0     0     0     0
      0     0     0     0     0     1     0     0     1     1     0
      -1    1     0     0     1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     -1    0     0     0
52    0     0     0     0     0     0     0     0     0     0     0
      0     13.0279     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     -1    0     0     0     0     -1    0
      1     0     -1    0     -1    0     0     0     0     0     0
      -1    0     1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     -1    0     0     0
53    0     0     0     0     0     0     0     0     0     0     0
      0     -13.0279    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0     0     0     0     1     0
      -1    0     1     0     1     0     0     0     0     0     0
      1     0     -1    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0     0     0
54    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     -1    0     1     1     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     1     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
55    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     -1    0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     -1    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0
56    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     -0.5  0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
```

```
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
57      0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0      -1       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0      -1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
58      0       0       0       0       0       0       0       0       0       0       0
        0      -1.7867          0      -1       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       1       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       1      -1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0
59      0       0       0       0       0       0       0       0       0       0       0
        0      -0.5191          0       0       0       0       1       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0      -1       0       0       0       0      -1       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0      -1       1       1       0      -1       0       0       0       0
        0       0       0       0       0       0       0       0
60      0       0       0       0       0       0       0       0       0       0       1
       -1      59.81    0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       1       0       0       0
        0       0       0       0      -1       0       1       1       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        1       1       0       1      -1       0       0       0       0       0       0
        0       1       0       0       0       0       0
61      0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0      -1       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
       -1       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0
62      0       0       0       0       0       0       0       0       0       0       0
        0      -2.8328          0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       1       0       0       0       0       1       0       0       0       0
        0       0       0       0       0       0       1       0       0       0       1
        1       0       0       0       0      -1       0      -1       0       0       0
        0       0       0       0      -1       0       1       1       0       0       0
        0       0       0       0       0       0       0       0
```

```
63    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0    -1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0    -1     0     0     0     0
      0     0     0     0     0     0     0
64    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     1
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
     -1     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
     -1     0     0     0     0     0     0
65    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0    -1     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0    -1
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      1     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      1     0     0     0     0     0     0
66    0     0     0     0     0     0     0     0     0     0     0
      0    -0.8977    0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0    -1     0
      0     0     0     0     0    -1     0     0
67    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     1     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0    -1     1     0     0
      0     0     0     0     0     0     0
68    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0    -1     0     1     0     0
69    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     0     0     0     0     1
      0     0     0     0     0     0     0     0     0     0     0
```

```
        0       0       0       0       1       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       1       -1
        -1      -1      0       0       0       0       0
70      0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       -1      0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       -1      1
        0       0       0       0       0       0       0
71      0       0       0       0       0       0       0       1       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       1       0       0       0       0       0
72      0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       1       0       0       0
        0       0       0       0       -1      -1      0;

param Vmin := 1  -1000
2       -1000
3       -1000
4       -1000
5       -1000
6       -1000
7       -1000
8       0
9       -1000
10      -1000
11      8.39
12      -1000
13      0
14      -1000
15      0
16      0
17      -1000
18      -1000
19      -1000
20      0
21      0
22      0
23      -1000
24      0
25      0
26      0
27      0
```

```
28    -10
29    0
30    0
31    -1000
32    -1000
33    0
34    0
35    -1000
36    0
37    -1000
38    0
39    0
40    -1000
41    0
42    0
43    0
44    0
45    0
46    -1000
47    0
48    -1000
49    -1000
50    0
51    0
52    0
53    -1000
54    0
55    0
56    -1000
57    0
58    -1000
59    -1000
60    0
61    -1000
62    0
63    0
64    -1000
65    0
66    0
67    0
68    0
69    -1000
70    -1000
71    0
72    0
73    0
74    -1000
75    -1000
76    0
77    -1000
78    -1000
79    0
80    0
81    0
82    -1000
83    0
84    -1000
85    -1000
86    -1000
```

```
87    0
88    0
89    0
90    -1000
91    -1000
92    0
93    -1000
94    -1000
95    -1000;

param Vmax := 1  1000
2     1000
3     1000
4     1000
5     1000
6     1000
7     1000
8     1000
9     1000
10    1000
11    1000
12    1000
13    1000
14    1000
15    1000
16    1000
17    1000
18    1000
19    1000
20    1000
21    1000
22    1000
23    1000
24    1000
25    1000
26    1000
27    1000
28    1000
29    1000
30    1000
31    1000
32    1000
33    1000
34    1000
35    1000
36    0
37    1000
38    1000
39    1000
40    1000
41    1000
42    1000
43    1000
44    1000
45    1000
46    1000
47    1000
48    1000
49    1000
```

```
50    1000
51    1000
52    1000
53    1000
54    1000
55    1000
56    1000
57    1000
58    1000
59    1000
60    1000
61    1000
62    1000
63    1000
64    1000
65    1000
66    1000
67    1000
68    1000
69    1000
70    1000
71    1000
72    1000
73    1000
74    1000
75    1000
76    1000
77    1000
78    1000
79    1000
80    1000
81    1000
82    1000
83    1000
84    1000
85    1000
86    1000
87    1000
88    1000
89    1000
90    1000
91    1000
92    1000
93    1000
94    1000
95    1000;

end;
```

Modelo core en formato COBRApy

```
#Modelo core
import cobra
from cobra import Model, Reaction, Metabolite
import numpy as np

model=Model('core_ptm')

reaction1 = Reaction('RXN1')
```

```
reaction2 = Reaction('RXN2')
reaction3 = Reaction('RXN3')
reaction4 = Reaction('RXN4')
reaction5 = Reaction('RXN5')
reaction6 = Reaction('RXN6')
reaction7 = Reaction('RXN7')
reaction8 = Reaction('RXN8')
reaction9 = Reaction('RXN9')
reaction10 = Reaction('RXN10')
reaction11 = Reaction('RXN11')
reaction12 = Reaction('RXN12')
reaction13 = Reaction('RXN13')
reaction14 = Reaction('RXN14')
reaction15 = Reaction('RXN15')
reaction16 = Reaction('RXN16')
reaction17 = Reaction('RXN17')
reaction18 = Reaction('RXN18')
reaction19 = Reaction('RXN19')
reaction20 = Reaction('RXN20')
reaction21 = Reaction('RXN21')
reaction22 = Reaction('RXN22')
reaction23 = Reaction('RXN23')
reaction24 = Reaction('RXN24')
reaction25 = Reaction('RXN25')
reaction26 = Reaction('RXN26')
reaction27 = Reaction('RXN27')
reaction28 = Reaction('RXN28')
reaction29 = Reaction('RXN29')
reaction30 = Reaction('RXN30')
reaction31 = Reaction('RXN31')
reaction32 = Reaction('RXN32')
reaction33 = Reaction('RXN33')
reaction34 = Reaction('RXN34')
reaction35 = Reaction('RXN35')
reaction36 = Reaction('RXN36')
reaction37 = Reaction('RXN37')
reaction38 = Reaction('RXN38')
reaction39 = Reaction('RXN39')
reaction40 = Reaction('RXN40')
reaction41 = Reaction('RXN41')
reaction42 = Reaction('RXN42')
reaction43 = Reaction('RXN43')
reaction44 = Reaction('RXN44')
reaction45 = Reaction('RXN45')
reaction46 = Reaction('RXN46')
reaction47 = Reaction('RXN47')
reaction48 = Reaction('RXN48')
reaction49 = Reaction('RXN49')
reaction50 = Reaction('RXN50')
reaction51 = Reaction('RXN51')
reaction52 = Reaction('RXN52')
reaction53 = Reaction('RXN53')
reaction54 = Reaction('RXN54')
reaction55 = Reaction('RXN55')
reaction56 = Reaction('RXN56')
reaction57 = Reaction('RXN57')
reaction58 = Reaction('RXN58')
reaction59 = Reaction('RXN59')
reaction60 = Reaction('RXN60')
```

```
reaction61 = Reaction('RXN61')
reaction62 = Reaction('RXN62')
reaction63 = Reaction('RXN63')
reaction64 = Reaction('RXN64')
reaction65 = Reaction('RXN65')
reaction66 = Reaction('RXN66')
reaction67 = Reaction('RXN67')
reaction68 = Reaction('RXN68')
reaction69 = Reaction('RXN69')
reaction70 = Reaction('RXN70')
reaction71 = Reaction('RXN71')
reaction72 = Reaction('RXN72')
reaction73 = Reaction('RXN73')
reaction74 = Reaction('RXN74')
reaction75 = Reaction('RXN75')
reaction76 = Reaction('RXN76')
reaction77 = Reaction('RXN77')
reaction78 = Reaction('RXN78')
reaction79 = Reaction('RXN79')
reaction80 = Reaction('RXN80')
reaction81 = Reaction('RXN81')
reaction82 = Reaction('RXN82')
reaction83 = Reaction('RXN83')
reaction84 = Reaction('RXN84')
reaction85 = Reaction('RXN85')
reaction86 = Reaction('RXN86')
reaction87 = Reaction('RXN87')
reaction88 = Reaction('RXN88')
reaction89 = Reaction('RXN89')
reaction90 = Reaction('RXN90')
reaction91 = Reaction('RXN91')
reaction92 = Reaction('RXN92')
reaction93 = Reaction('RXN93')
reaction94 = Reaction('RXN94')
reaction95 = Reaction('RXN95')


reaction1.lower_bound = -1000
reaction2.lower_bound = -1000
reaction3.lower_bound = -1000
reaction4.lower_bound = -1000
reaction5.lower_bound = -1000
reaction6.lower_bound = -1000
reaction7.lower_bound = -1000
reaction8.lower_bound = 0
reaction9.lower_bound = -1000
reaction10.lower_bound = -1000
reaction11.lower_bound = 8.39
reaction12.lower_bound = -1000
reaction13.lower_bound = 0
reaction14.lower_bound = -1000
reaction15.lower_bound = 0
reaction16.lower_bound = 0
reaction17.lower_bound = -1000
reaction18.lower_bound = -1000
reaction19.lower_bound = -1000
reaction20.lower_bound = 0
reaction21.lower_bound = 0
reaction22.lower_bound = 0
```

```
reaction23.lower_bound = -1000
reaction24.lower_bound = 0
reaction25.lower_bound = 0
reaction26.lower_bound = 0
reaction27.lower_bound = 0
reaction28.lower_bound = -10
reaction29.lower_bound = 0
reaction30.lower_bound = 0
reaction31.lower_bound = -1000
reaction32.lower_bound = -1000
reaction33.lower_bound = 0
reaction34.lower_bound = 0
reaction35.lower_bound = -1000
reaction36.lower_bound = -1000
reaction37.lower_bound = -1000
reaction38.lower_bound = 0
reaction39.lower_bound = 0
reaction40.lower_bound = -1000
reaction41.lower_bound = 0
reaction42.lower_bound = 0
reaction43.lower_bound = 0
reaction44.lower_bound = 0
reaction45.lower_bound = 0
reaction46.lower_bound = -1000
reaction47.lower_bound = 0
reaction48.lower_bound = -1000
reaction49.lower_bound = -1000
reaction50.lower_bound = 0
reaction51.lower_bound = 0
reaction52.lower_bound = 0
reaction53.lower_bound = -1000
reaction54.lower_bound = 0
reaction55.lower_bound = 0
reaction56.lower_bound = -1000
reaction57.lower_bound = 0
reaction58.lower_bound = -1000
reaction59.lower_bound = -1000
reaction60.lower_bound = 0
reaction61.lower_bound = -1000
reaction62.lower_bound = 0
reaction63.lower_bound = 0
reaction64.lower_bound = -1000
reaction65.lower_bound = 0
reaction66.lower_bound = 0
reaction67.lower_bound = 0
reaction68.lower_bound = 0
reaction69.lower_bound = -1000
reaction70.lower_bound = -1000
reaction71.lower_bound = 0
reaction72.lower_bound = 0
reaction73.lower_bound = 0
reaction74.lower_bound = -1000
reaction75.lower_bound = -1000
reaction76.lower_bound = 0
reaction77.lower_bound = -1000
reaction78.lower_bound = -1000
reaction79.lower_bound = 0
reaction80.lower_bound = 0
reaction81.lower_bound = 0
```

```
reaction82.lower_bound = -1000
reaction83.lower_bound = 0
reaction84.lower_bound = -1000
reaction85.lower_bound = -1000
reaction86.lower_bound = -1000
reaction87.lower_bound = 0
reaction88.lower_bound = 0
reaction89.lower_bound = 0
reaction90.lower_bound = -1000
reaction91.lower_bound = -1000
reaction92.lower_bound = 0
reaction93.lower_bound = -1000
reaction94.lower_bound = -1000
reaction95.lower_bound = -1000

reaction1.upper_bound = 1000
reaction2.upper_bound = 1000
reaction3.upper_bound = 1000
reaction4.upper_bound = 1000
reaction5.upper_bound = 1000
reaction6.upper_bound = 1000
reaction7.upper_bound = 1000
reaction8.upper_bound = 1000
reaction9.upper_bound = 1000
reaction10.upper_bound = 1000
reaction11.upper_bound = 1000
reaction12.upper_bound = 1000
reaction13.upper_bound = 1000
reaction14.upper_bound = 1000
reaction15.upper_bound = 1000
reaction16.upper_bound = 1000
reaction17.upper_bound = 1000
reaction18.upper_bound = 1000
reaction19.upper_bound = 1000
reaction20.upper_bound = 1000
reaction21.upper_bound = 1000
reaction22.upper_bound = 1000
reaction23.upper_bound = 1000
reaction24.upper_bound = 1000
reaction25.upper_bound = 1000
reaction26.upper_bound = 1000
reaction27.upper_bound = 1000
reaction28.upper_bound = 1000
reaction29.upper_bound = 1000
reaction30.upper_bound = 1000
reaction31.upper_bound = 1000
reaction32.upper_bound = 1000
reaction33.upper_bound = 1000
reaction34.upper_bound = 1000
reaction35.upper_bound = 1000
reaction36.upper_bound = 1000
reaction37.upper_bound = 1000
reaction38.upper_bound = 1000
reaction39.upper_bound = 1000
reaction40.upper_bound = 1000
reaction41.upper_bound = 1000
reaction42.upper_bound = 1000
reaction43.upper_bound = 1000
reaction44.upper_bound = 1000
```

```
reaction45.upper_bound = 1000
reaction46.upper_bound = 1000
reaction47.upper_bound = 1000
reaction48.upper_bound = 1000
reaction49.upper_bound = 1000
reaction50.upper_bound = 1000
reaction51.upper_bound = 1000
reaction52.upper_bound = 1000
reaction53.upper_bound = 1000
reaction54.upper_bound = 1000
reaction55.upper_bound = 1000
reaction56.upper_bound = 1000
reaction57.upper_bound = 1000
reaction58.upper_bound = 1000
reaction59.upper_bound = 1000
reaction60.upper_bound = 1000
reaction61.upper_bound = 1000
reaction62.upper_bound = 1000
reaction63.upper_bound = 1000
reaction64.upper_bound = 1000
reaction65.upper_bound = 1000
reaction66.upper_bound = 1000
reaction67.upper_bound = 1000
reaction68.upper_bound = 1000
reaction69.upper_bound = 1000
reaction70.upper_bound = 1000
reaction71.upper_bound = 1000
reaction72.upper_bound = 1000
reaction73.upper_bound = 1000
reaction74.upper_bound = 1000
reaction75.upper_bound = 1000
reaction76.upper_bound = 1000
reaction77.upper_bound = 1000
reaction78.upper_bound = 1000
reaction79.upper_bound = 1000
reaction80.upper_bound = 1000
reaction81.upper_bound = 1000
reaction82.upper_bound = 1000
reaction83.upper_bound = 1000
reaction84.upper_bound = 1000
reaction85.upper_bound = 1000
reaction86.upper_bound = 1000
reaction87.upper_bound = 1000
reaction88.upper_bound = 1000
reaction89.upper_bound = 1000
reaction90.upper_bound = 1000
reaction91.upper_bound = 1000
reaction92.upper_bound = 1000
reaction93.upper_bound = 1000
reaction94.upper_bound = 1000
reaction95.upper_bound = 1000

MET1=Metabolite('MET1')
MET2=Metabolite('MET2')
MET3=Metabolite('MET3')
MET4=Metabolite('MET4')
MET5=Metabolite('MET5')
MET6=Metabolite('MET6')
MET7=Metabolite('MET7')
```

```
MET8=Metabolite('MET8')
MET9=Metabolite('MET9')
MET10=Metabolite('MET10')
MET11=Metabolite('MET11')
MET12=Metabolite('MET12')
MET13=Metabolite('MET13')
MET14=Metabolite('MET14')
MET15=Metabolite('MET15')
MET16=Metabolite('MET16')
MET17=Metabolite('MET17')
MET18=Metabolite('MET18')
MET19=Metabolite('MET19')
MET20=Metabolite('MET20')
MET21=Metabolite('MET21')
MET22=Metabolite('MET22')
MET23=Metabolite('MET23')
MET24=Metabolite('MET24')
MET25=Metabolite('MET25')
MET26=Metabolite('MET26')
MET27=Metabolite('MET27')
MET28=Metabolite('MET28')
MET29=Metabolite('MET29')
MET30=Metabolite('MET30')
MET31=Metabolite('MET31')
MET32=Metabolite('MET32')
MET33=Metabolite('MET33')
MET34=Metabolite('MET34')
MET35=Metabolite('MET35')
MET36=Metabolite('MET36')
MET37=Metabolite('MET37')
MET38=Metabolite('MET38')
MET39=Metabolite('MET39')
MET40=Metabolite('MET40')
MET41=Metabolite('MET41')
MET42=Metabolite('MET42')
MET43=Metabolite('MET43')
MET44=Metabolite('MET44')
MET45=Metabolite('MET45')
MET46=Metabolite('MET46')
MET47=Metabolite('MET47')
MET48=Metabolite('MET48')
MET49=Metabolite('MET49')
MET50=Metabolite('MET50')
MET51=Metabolite('MET51')
MET52=Metabolite('MET52')
MET53=Metabolite('MET53')
MET54=Metabolite('MET54')
MET55=Metabolite('MET55')
MET56=Metabolite('MET56')
MET57=Metabolite('MET57')
MET58=Metabolite('MET58')
MET59=Metabolite('MET59')
MET60=Metabolite('MET60')
MET61=Metabolite('MET61')
MET62=Metabolite('MET62')
MET63=Metabolite('MET63')
MET64=Metabolite('MET64')
MET65=Metabolite('MET65')
MET66=Metabolite('MET66')
```

```
MET67=Metabolite('MET67')
MET68=Metabolite('MET68')
MET69=Metabolite('MET69')
MET70=Metabolite('MET70')
MET71=Metabolite('MET71')
MET72=Metabolite('MET72')


reaction1.add_metabolites({MET8:-1,MET10:1,MET21:-1,MET43:1,MET50:-
1,MET51:1})
reaction2.add_metabolites({MET8:1,MET9:-1})
reaction3.add_metabolites({MET6:-1,MET12:1,MET13:1,MET17:-1})
reaction4.add_metabolites({MET11:1,MET18:-1,MET41:1})
reaction5.add_metabolites({MET11:-1,MET41:-1,MET45:1})
reaction6.add_metabolites({MET6:1,MET7:-1,MET43:1,MET44:-1})
reaction7.add_metabolites({MET13:2,MET16:-1,MET17:-1})
reaction8.add_metabolites({MET14:-1,MET19:1,MET21:-1,MET50:-
1,MET51:1,MET71:1})
reaction9.add_metabolites({MET14:1,MET15:-1,MET43:1,MET44:-1})
reaction10.add_metabolites({MET8:1,MET24:-1,MET43:1,MET50:-1,MET51:1})
reaction11.add_metabolites({MET13:1,MET17:-1,MET41:-1,MET43:1,MET60:1})
reaction12.add_metabolites({MET13:-1,MET17:1,MET41:1,MET43:3,MET44:-4,MET60:-
1})
reaction13.add_metabolites({MET3:-1.496,MET10:-
3.7478,MET13:59.81,MET14:4.1182,MET17:-59.81,MET21:3.7478,MET23:-
0.361,MET26:-0.0709,MET33:-0.129,MET34:-0.205,MET36:-0.2557,MET38:-
4.9414,MET41:-59.81,MET43:59.81,MET50:-
3.547,MET51:3.547,MET52:13.0279,MET53:-13.0279,MET58:-1.7867,MET59:-
0.5191,MET60:59.81,MET62:-2.8328,MET66:-0.8977})
reaction14.add_metabolites({MET19:1,MET20:-1})
reaction15.add_metabolites({MET10:-1,MET18:1,MET21:1,MET41:-1,MET43:1,MET58:-
1})
reaction16.add_metabolites({MET41:1,MET43:-2,MET44:2,MET56:-
0.5,MET64:1,MET65:-1})
reaction17.add_metabolites({MET43:1,MET44:-1,MET46:1,MET47:-1})
reaction18.add_metabolites({MET2:-1,MET41:1,MET59:1})
reaction19.add_metabolites({MET24:1,MET25:-1,MET43:1,MET44:-1})
reaction20.add_metabolites({MET7:-1})
reaction21.add_metabolites({MET9:-1})
reaction22.add_metabolites({MET15:-1})
reaction23.add_metabolites({MET20:-1})
reaction24.add_metabolites({MET25:-1})
reaction25.add_metabolites({MET29:-1})
reaction26.add_metabolites({MET30:-1})
reaction27.add_metabolites({MET32:-1})
reaction28.add_metabolites({MET35:-1})
reaction29.add_metabolites({MET37:-1})
reaction30.add_metabolites({MET39:-1})
reaction31.add_metabolites({MET44:-1})
reaction32.add_metabolites({MET42:-1})
reaction33.add_metabolites({MET47:-1})
reaction34.add_metabolites({MET49:-1})
reaction35.add_metabolites({MET55:-1})
reaction36.add_metabolites({MET57:-1})
reaction37.add_metabolites({MET61:-1})
reaction38.add_metabolites({MET63:-1})
reaction39.add_metabolites({MET70:-1})
reaction40.add_metabolites({MET22:1,MET27:-1,MET33:1})
reaction41.add_metabolites({MET26:1,MET27:-1,MET41:-1,MET60:1})
```

```
reaction42.add_metabolites({MET28:1,MET29:-1,MET43:1,MET44:-1})
reaction43.add_metabolites({MET28:-1,MET29:1})
reaction44.add_metabolites({MET31:-1,MET64:1,MET65:-1,MET69:1})
reaction45.add_metabolites({MET26:1,MET30:-1,MET59:-1,MET62:1})
reaction46.add_metabolites({MET31:-1,MET41:-1,MET48:1})
reaction47.add_metabolites({MET31:1,MET32:-1,MET43:2,MET44:-2})
reaction48.add_metabolites({MET5:1,MET34:-1,MET43:1,MET52:-1,MET53:1})
reaction49.add_metabolites({MET1:1,MET33:-1,MET43:1,MET50:-1,MET51:1,MET60:-
1})
reaction50.add_metabolites({MET34:1,MET35:-1,MET59:-1,MET62:1})
reaction51.add_metabolites({MET13:1,MET17:-1,MET36:1,MET38:-1,MET43:1,MET54:-
1,MET60:1})
reaction52.add_metabolites({MET13:1,MET17:-1,MET36:1,MET37:-1,MET41:-
1,MET43:1,MET60:1})
reaction53.add_metabolites({MET14:1,MET38:-1,MET41:-1,MET43:1,MET52:-
1,MET53:1,MET54:1})
reaction54.add_metabolites({MET36:-1,MET38:1,MET41:-1,MET54:1})
reaction55.add_metabolites({MET14:-1,MET36:-1,MET38:2,MET43:-
1,MET52:1,MET53:-1})
reaction56.add_metabolites({MET38:1,MET39:-1,MET43:1,MET44:-1})
reaction57.add_metabolites({MET4:-1,MET19:1,MET52:-1,MET53:1,MET67:1})
reaction58.add_metabolites({MET41:1,MET42:-1})
reaction59.add_metabolites({MET14:1,MET19:1,MET45:-1,MET52:-1,MET53:1})
reaction60.add_metabolites({MET40:1,MET45:-1,MET69:1})
reaction61.add_metabolites({MET43:1,MET46:-1,MET50:-1,MET51:1,MET62:1})
reaction62.add_metabolites({MET10:-1,MET21:1,MET40:-1,MET41:-
1,MET43:1,MET48:1})
reaction63.add_metabolites({MET43:2,MET44:-2,MET48:1,MET49:-1})
reaction64.add_metabolites({MET43:1,MET48:-1,MET50:-1,MET51:1,MET58:1})
reaction65.add_metabolites({MET19:1,MET48:-1,MET50:-1,MET51:1,MET62:1})
reaction66.add_metabolites({MET19:1,MET48:-1,MET52:-1,MET53:1,MET62:1})
reaction67.add_metabolites({MET43:-4,MET44:3,MET50:1,MET51:-1,MET64:-
1,MET65:1})
reaction68.add_metabolites({MET50:-1,MET51:1,MET52:1,MET53:-1})
reaction69.add_metabolites({MET54:1,MET55:-1})
reaction70.add_metabolites({MET56:1,MET57:-1})
reaction71.add_metabolites({MET10:1,MET19:1,MET21:-1,MET50:-1,MET51:1,MET62:-
1})
reaction72.add_metabolites({MET13:1,MET17:-1,MET26:-1,MET27:1,MET43:1})
reaction73.add_metabolites({MET10:1,MET21:-1,MET28:1,MET62:-1})
reaction74.add_metabolites({MET26:1,MET34:-1})
reaction75.add_metabolites({MET1:1,MET3:-1,MET13:1,MET17:-1})
reaction76.add_metabolites({MET4:1,MET5:-1,MET41:-1,MET43:1})
reaction77.add_metabolites({MET2:-1,MET3:1})
reaction78.add_metabolites({MET43:1,MET44:-1,MET60:1,MET61:-1})
reaction79.add_metabolites({MET19:-1,MET41:-1,MET43:1,MET58:1,MET59:-
1,MET60:1})
reaction80.add_metabolites({MET13:1,MET17:-1,MET19:1,MET58:-1,MET59:1})
reaction81.add_metabolites({MET16:1,MET17:-1,MET41:-
1,MET43:2,MET59:1,MET60:1,MET62:-1})
reaction82.add_metabolites({MET10:-1,MET12:1,MET21:1,MET60:-1})
reaction83.add_metabolites({MET13:-1,MET17:1,MET43:-1,MET59:-1,MET62:1})
reaction84.add_metabolites({MET43:1,MET44:-1,MET62:1,MET63:-1})
reaction85.add_metabolites({MET67:-1,MET72:1})
reaction86.add_metabolites({MET66:-1,MET67:1})
reaction87.add_metabolites({MET43:2,MET44:-2,MET69:1,MET70:-1})
reaction88.add_metabolites({MET43:1,MET44:-1,MET69:-1,MET70:1})
reaction89.add_metabolites({MET31:1,MET64:-1,MET65:1,MET69:-1})
```

```
reaction90.add_metabolites({MET13:1,MET17:-1,MET21:-1,MET60:1,MET69:-
1,MET71:1})
reaction91.add_metabolites({MET23:1,MET26:1,MET33:-1,MET68:-1})
reaction92.add_metabolites({MET43:2,MET44:-2,MET50:1,MET51:-1,MET52:-
1,MET53:1})
reaction93.add_metabolites({MET33:1,MET66:-1,MET68:1,MET72:-1})
reaction94.add_metabolites({MET23:-1,MET26:1,MET33:1,MET72:-1})
reaction95.add_metabolites({MET22:-1,MET33:1})


model.add_reactions([reaction1])
model.add_reactions([reaction2])
model.add_reactions([reaction3])
model.add_reactions([reaction4])
model.add_reactions([reaction5])
model.add_reactions([reaction6])
model.add_reactions([reaction7])
model.add_reactions([reaction8])
model.add_reactions([reaction9])
model.add_reactions([reaction10])
model.add_reactions([reaction11])
model.add_reactions([reaction12])
model.add_reactions([reaction13])
model.add_reactions([reaction14])
model.add_reactions([reaction15])
model.add_reactions([reaction16])
model.add_reactions([reaction17])
model.add_reactions([reaction18])
model.add_reactions([reaction19])
model.add_reactions([reaction20])
model.add_reactions([reaction21])
model.add_reactions([reaction22])
model.add_reactions([reaction23])
model.add_reactions([reaction24])
model.add_reactions([reaction25])
model.add_reactions([reaction26])
model.add_reactions([reaction27])
model.add_reactions([reaction28])
model.add_reactions([reaction29])
model.add_reactions([reaction30])
model.add_reactions([reaction31])
model.add_reactions([reaction32])
model.add_reactions([reaction33])
model.add_reactions([reaction34])
model.add_reactions([reaction35])
model.add_reactions([reaction36])
model.add_reactions([reaction37])
model.add_reactions([reaction38])
model.add_reactions([reaction39])
model.add_reactions([reaction40])
model.add_reactions([reaction41])
model.add_reactions([reaction42])
model.add_reactions([reaction43])
model.add_reactions([reaction44])
model.add_reactions([reaction45])
model.add_reactions([reaction46])
model.add_reactions([reaction47])
model.add_reactions([reaction48])
model.add_reactions([reaction49])
```

```
model.add_reactions([reaction50])
model.add_reactions([reaction51])
model.add_reactions([reaction52])
model.add_reactions([reaction53])
model.add_reactions([reaction54])
model.add_reactions([reaction55])
model.add_reactions([reaction56])
model.add_reactions([reaction57])
model.add_reactions([reaction58])
model.add_reactions([reaction59])
model.add_reactions([reaction60])
model.add_reactions([reaction61])
model.add_reactions([reaction62])
model.add_reactions([reaction63])
model.add_reactions([reaction64])
model.add_reactions([reaction65])
model.add_reactions([reaction66])
model.add_reactions([reaction67])
model.add_reactions([reaction68])
model.add_reactions([reaction69])
model.add_reactions([reaction70])
model.add_reactions([reaction71])
model.add_reactions([reaction72])
model.add_reactions([reaction73])
model.add_reactions([reaction74])
model.add_reactions([reaction75])
model.add_reactions([reaction76])
model.add_reactions([reaction77])
model.add_reactions([reaction78])
model.add_reactions([reaction79])
model.add_reactions([reaction80])
model.add_reactions([reaction81])
model.add_reactions([reaction82])
model.add_reactions([reaction83])
model.add_reactions([reaction84])
model.add_reactions([reaction85])
model.add_reactions([reaction86])
model.add_reactions([reaction87])
model.add_reactions([reaction88])
model.add_reactions([reaction89])
model.add_reactions([reaction90])
model.add_reactions([reaction91])
model.add_reactions([reaction92])
model.add_reactions([reaction93])
model.add_reactions([reaction94])
model.add_reactions([reaction95])
```

Anexo 10: Modelo E.coli compact core en formato GLPK y COBRApy

Modelo compact core en formato GLPK

```
param S{j in 1..13, i in 1..25};
param Vmax{i in 1..25};
param Vmin{i in 1..25};
var v{i in 1..25};

maximize z: v[19];

s.t. balance{j in 1..13}: sum{i in 1..25} S[j,i]*v[i]=0;
s.t. upperbound{i in 1..25}: v[i] <= Vmax[i];
s.t. lowerbound{i in 1..25}: v[i] >= Vmin[i];

solve;
printf{i in 1..25}: "%s\n",v[i];

data;

param S : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25:=
1     0     0     1     -1    0     0     0     0     -1    -1    1
      0     0     0     0     0     0     0     0     0     0     0
      0     0     -3.7478
2     -1    -1    0     0     0     1     0     0     0     -1    0
      0     0     -1    0     0     0     0     1     0     -1    -1.25
      0     0     59.81
3     0     0     0     1     -1    0     -1    0     0     0     0
      -1    0     -1    0     0     0     0     0     0     0     0
      0     0     4.1182
4     0     0     0     0     0     0     0     0     0     0     0
      0     0     0     1     -1    0     0     0     0     0     0
      -1    0     0
5     0     0     0     0     0     1     -1    0     0     0     0
      0     0     0     0     0     0     0     0     0     0     0
      0     0     -0.2557
6     0     0     0     0     1     -1    2     0     0     0     0
      0     -1    0     0     0     0     0     0     0     0     0
      0     0     -4.9414
7     -2    0     -1    0     0     0     0     1     2     0     0
      0     0     -1    0     -1    0     0     0     -1    0     1
      0     1     -3.547
8     0     0     0     1     -1    0     -1    0     0     0     0
      0     0     0     0     0     0     0     0     -1    0     0
      0     0     -13.0279
9     0     0     0     -1    0     0     0     0     0     0     0
      0     0     0     0     1     0     1     0     0     0     0
      0     0     -1.7867
10    1     -1    0     0     0     0     0     0     0     0     0
      0     0     0     0     0     0     -1    0     0     0     0
      0     0     -0.5191
11    1     1     -1    0     0     0     0     -1    0     0     -1
      0     0     0     0     0     0     0     0     0     0     0
      0     0     -2.8328
12    0     0     0     0     0     0     0     0     0     0     0
      0     0     0     -1    0     0     0     0     0     2     0
      1     -1    0
```

```
13    0     0     0     0     0     0     0     0     0     0
      0     0     1     -1    0     -1    0     0     0     0
      1     0     0;


param Vmax := 1  10
2   1000
3   1000
4   1000
5   1000
6   1000
7   1000
8   1000
9   1000
10   1000
11   1000
12   1000
13   1000
14   1000
15   0
16   1000
17   1000
18   1000
19   1000
20   1000
21   0
22   0
23   1000
24   1000
25   1000;

param Vmin := 1  0
2   0
3   0
4   0
5   -1000
6   0
7   0
8   0
9   0
10   0
11   0
12   0
13   0
14   0
15   0
16   -1000
17   0
18   0
19   8.39
20   -1000
21   0
22   0
23   0
24   0
25   0;

end;
```

Modelo compact core en formato COBRApy

```
import cobra
from cobra import Model, Reaction, Metabolite
import numpy as np
model=Model('juguete_20180312')

reaction1=Reaction('V01')
reaction2=Reaction('V02')
reaction3=Reaction('V03')
reaction4=Reaction('V04')
reaction5=Reaction('V05')
reaction6=Reaction('V06')
reaction7=Reaction('V07')
reaction8=Reaction('V08')
reaction9=Reaction('V09')
reaction10=Reaction('V10')
reaction11=Reaction('V11')
reaction12=Reaction('V12')
reaction13=Reaction('V13')
reaction14=Reaction('V14')
reaction15=Reaction('V15')
reaction16=Reaction('V16')
reaction17=Reaction('V17')
reaction18=Reaction('V18')
reaction19=Reaction('V19')
reaction20=Reaction('V20')
reaction21=Reaction('V22')
reaction22=Reaction('V23')
reaction23=Reaction('V24')
reaction24=Reaction('V25')
reaction25=Reaction('VBM')

reaction1.lower_bound=0
reaction2.lower_bound=0
reaction3.lower_bound=0
reaction4.lower_bound=0
reaction5.lower_bound=-1000
reaction6.lower_bound=0
reaction7.lower_bound=0
reaction8.lower_bound=0
reaction9.lower_bound=0
reaction10.lower_bound=0
reaction11.lower_bound=0
reaction12.lower_bound=0
reaction13.lower_bound=0
reaction14.lower_bound=0
reaction15.lower_bound=0
reaction16.lower_bound=-1000
reaction17.lower_bound=0
reaction18.lower_bound=0
reaction19.lower_bound=8.39
reaction20.lower_bound=-1000
reaction21.lower_bound=0
reaction22.lower_bound=0
reaction23.lower_bound=0
reaction24.lower_bound=0
reaction25.lower_bound=0
```

```
reaction1.upper_bound=10
reaction2.upper_bound=1000
reaction3.upper_bound=1000
reaction4.upper_bound=1000
reaction5.upper_bound=1000
reaction6.upper_bound=1000
reaction7.upper_bound=1000
reaction8.upper_bound=1000
reaction9.upper_bound=1000
reaction10.upper_bound=1000
reaction11.upper_bound=1000
reaction12.upper_bound=1000
reaction13.upper_bound=1000
reaction14.upper_bound=1000
reaction15.upper_bound=0
reaction16.upper_bound=1000
reaction17.upper_bound=1000
reaction18.upper_bound=1000
reaction19.upper_bound=1000
reaction20.upper_bound=1000
reaction21.upper_bound=0
reaction22.upper_bound=0
reaction23.upper_bound=1000
reaction24.upper_bound=1000
reaction25.upper_bound=1000

ACoA=Metabolite('ACoA')
ADP=Metabolite('ADP')
aKG=Metabolite('aKG')
ATP=Metabolite('ATP')
FUM=Metabolite('FUM')
GLN=Metabolite('GLN')
GLU=Metabolite('GLU')
NAD=Metabolite('NAD')
NADH=Metabolite('NADH')
NADP=Metabolite('NADP')
NADPH=Metabolite('NADPH')
OAA=Metabolite('OAA')
PEP=Metabolite('PEP')
PYR=Metabolite('PYR')
q8=Metabolite('q8')
q8h2=Metabolite('q8h2')
SUCC=Metabolite('SUCC')

reaction1.add_metabolites({ADP:-1,ATP:1,NAD:-2,NADH:2,PEP:1,PYR:1})
reaction2.add_metabolites({ADP:-1,ATP:1,PEP:-1,PYR:1})
reaction3.add_metabolites({ACoA:1,NAD:-1,NADH:1,PYR:-1})
reaction4.add_metabolites({ACoA:-1,aKG:1,NADP:-1,NADPH:1,OAA:-1})
reaction5.add_metabolites({aKG:-1,GLU:1,NADP:1,NADPH:-1})
reaction6.add_metabolites({ADP:1,ATP:-1,GLN:1,GLU:-1})
reaction7.add_metabolites({aKG:-1,GLN:-1,GLU:2,NADP:1,NADPH:-1})
reaction8.add_metabolites({NAD:1,NADH:-1,PYR:-1})
reaction9.add_metabolites({ACoA:-1,NAD:2,NADH:-2})
reaction10.add_metabolites({ACoA:-1,ADP:-1,ATP:1})
reaction11.add_metabolites({ACoA:1,PYR:-1})
reaction12.add_metabolites({aKG:-1})
reaction13.add_metabolites({GLU:-1})
reaction14.add_metabolites({ADP:-1,aKG:-1,ATP:1,NAD:-1,NADH:1,SUCC:1})
reaction15.add_metabolites({FUM:1,q8:-1,q8h2:1,SUCC:-1})
```

```
reaction16.add_metabolites({FUM:-1,NAD:-1,NADH:1,OAA:1})
reaction17.add_metabolites({SUCC:-1})
reaction18.add_metabolites({OAA:1,PEP:-1})
reaction19.add_metabolites({ADP:1,ATP:-1})
reaction20.add_metabolites({NAD:-1,NADH:1,NADP:1,NADPH:-1})
reaction21.add_metabolites({ADP:-1,ATP:1,q8:2,q8h2:-2})
reaction22.add_metabolites({ADP:-1.25,ATP:1.25,NAD:1,NADH:-1})
reaction23.add_metabolites({FUM:-1,q8:1,q8h2:-1,SUCC:1})
reaction24.add_metabolites({NAD:1,NADH:-1,q8:-1,q8h2:1})
reaction25.add_metabolites({ACoA:-3.7478,ADP:59.81,aKG:4.1182,ATP:-
59.81,GLN:-0.2557,GLU:-4.9414,NAD:-3.547,NADH:3.547,NADP:13.0279,NADPH:-
13.0279,OAA:-1.7867,PEP:-0.5191,PYR:-2.8328})

model.add_reactions([reaction1])
model.add_reactions([reaction2])
model.add_reactions([reaction3])
model.add_reactions([reaction4])
model.add_reactions([reaction5])
model.add_reactions([reaction6])
model.add_reactions([reaction7])
model.add_reactions([reaction8])
model.add_reactions([reaction9])
model.add_reactions([reaction10])
model.add_reactions([reaction11])
model.add_reactions([reaction12])
model.add_reactions([reaction13])
model.add_reactions([reaction14])
model.add_reactions([reaction15])
model.add_reactions([reaction16])
model.add_reactions([reaction17])
model.add_reactions([reaction18])
model.add_reactions([reaction19])
model.add_reactions([reaction20])
model.add_reactions([reaction21])
model.add_reactions([reaction22])
model.add_reactions([reaction23])
model.add_reactions([reaction24])
model.add_reactions([reaction25])
```

Anexo 11: Generación de cajas de muestreo en modelo E.coli core

```
import numpy
import scipy
from scipy import linalg
import os
import shutil
import fileinput
import sys

def caja(a_elim):
    #Cargar el modelo compact core original

sto_matrix_input=numpy.loadtxt(open("sto_matrix.csv","rb"),delimiter=";",skip
rows=1)

flujos_maximos_input=numpy.loadtxt(open("flujos_maximos.csv","rb"),delimiter=
";",skiprows=1)

flujos_minimos_input=numpy.loadtxt(open("flujos_minimos.csv","rb"),delimiter=
";",skiprows=1)

    #Eliminar los flujos que no van
    S=numpy.delete(sto_matrix_input,a_elim,1)
    maximos=numpy.delete(flujos_maximos_input,a_elim,0)
    minimos=numpy.delete(flujos_minimos_input,a_elim,0)

    #Rango
    rango=numpy.linalg.matrix_rank(S)

    #Dimension del espacio de soluciones
    dimension=S.shape[1]-rango

    #Descomposicion QR a partir de la matriz estequiometrica transpuesta
    St=numpy.matrix.transpose(S)
    Q,R=scipy.linalg.qr(St)

    #Definir matriz W
    W=Q[:,Q.shape[1]-dimension::1]

    #Generar data para GLPK
    f=open("auxiliar_data.dat","w")
    f.write("param w_filas := ")
    f.write(str(W.shape[0]))
    f.write(";\n")
    f.write("param w_columnas := ")
    f.write(str(W.shape[1]))
    f.write(";\n")
    f.write("param W : ")
    for i in range (0,dimension):
        f.write(str(i+1))
        f.write(" ")
    f.write(":=\n")
    for i in range (0,W.shape[0]):
        f.write(str(i+1))
        f.write(" ")
        for j in range (0,W.shape[1]):
            f.write(str(W[i,j]))
            f.write(" ")
```

```
    if i==W.shape[0]-1:
        f.write(";\n")
    else:
        f.write("\n")
f.write("\n")
f.write("param Vmax := ")
for i in range (0,maximos.shape[0]):
    f.write(str(i+1))
    f.write(" ")
    f.write(str(maximos[i]))
    if i==maximos.shape[0]-1:
        f.write(";\n")
    else:
        f.write("\n")
f.write("\n")
f.write("param Vmin := ")
for i in range (0,minimos.shape[0]):
    f.write(str(i+1))
    f.write(" ")
    f.write(str(minimos[i]))
    if i==minimos.shape[0]-1:
        f.write(";\n")
    else:
        f.write("\n")
f.write("\n")
f.write("end;")
f.close()

#Generar modelos GLPK para caja
#1 - Copiar el modelo base
for i in range (0,dimension):
    auxnombretexto=["auxiliar_caja_max_",str(i+1),".mod"]
    nombretexto="".join(auxnombretexto)
    shutil.copyfile("glpk_caja_base.mod",nombretexto)
for i in range (0,dimension):
    auxnombretexto=["auxiliar_caja_min_",str(i+1),".mod"]
    nombretexto="".join(auxnombretexto)
    shutil.copyfile("glpk_caja_base.mod",nombretexto)
#2 - Reemplazar en las copias
for i in range (0,dimension):
    auxnombretexto=["auxiliar_caja_max_",str(i+1),".mod"]
    nombretexto="".join(auxnombretexto)
    for line in fileinput.input(nombretexto,inplace=1):
        if "replace_maxmin" in line:
            line = line.replace("replace_maxmin","maximize")
        sys.stdout.write(line)
    for line in fileinput.input(nombretexto,inplace=1):
        if "replace_x" in line:
            line = line.replace("replace_x",str(i+1))
        sys.stdout.write(line)
for i in range (0,dimension):
    auxnombretexto=["auxiliar_caja_min_",str(i+1),".mod"]
    nombretexto="".join(auxnombretexto)
    for line in fileinput.input(nombretexto,inplace=1):
        if "replace_maxmin" in line:
            line = line.replace("replace_maxmin","minimize")
        sys.stdout.write(line)
    for line in fileinput.input(nombretexto,inplace=1):
        if "replace_x" in line:
```

```
                line = line.replace("replace_x",str(i+1))
            sys.stdout.write(line)

    #Correr GLPK para caja
    for i in range (0,dimension):
        auxnombretexto=["glpsol.exe                                    --model
auxiliar_caja_max_",str(i+1),".mod    --data    auxiliar_data.dat    --display
auxiliar_caja_max_",str(i+1),".csv"]
        nombretexto="".join(auxnombretexto)
        os.system(nombretexto)
    for i in range (0,dimension):
        auxnombretexto=["glpsol.exe                                    --model
auxiliar_caja_min_",str(i+1),".mod    --data    auxiliar_data.dat    --display
auxiliar_caja_min_",str(i+1),".csv"]
        nombretexto="".join(auxnombretexto)
        os.system(nombretexto)

    #Leer resultados para determinar caja
    cotas_sup=[]
    cotas_inf=[]
    for i in range (0,dimension):
        auxnombretexto=["auxiliar_caja_max_",str(i+1),".csv"]
        nombretexto="".join(auxnombretexto)

puntos_max=numpy.loadtxt(open(nombretexto,"rb"),delimiter=";",skiprows=0)
        auxnombretexto=["auxiliar_caja_min_",str(i+1),".csv"]
        nombretexto="".join(auxnombretexto)

puntos_min=numpy.loadtxt(open(nombretexto,"rb"),delimiter=";",skiprows=0)
        cotas_sup.append(puntos_max[i])
        cotas_inf.append(puntos_min[i])

    #Volumen de la caja
    vol_caja=numpy.prod(numpy.array(cotas_sup)-numpy.array(cotas_inf))

    #Exportar los resultados a un nuevo directorio
    auxnombretexto=["archivos_eliminando_rxns_",str(a_elim)]
    nombretexto="".join(auxnombretexto)
    current=os.getcwd()
    if not os.path.exists(nombretexto):
        os.mkdir(nombretexto)
        os.chdir(nombretexto)
        numpy.savetxt("flujos_a_elim.csv",a_elim,delimiter=";",newline="\n")
        numpy.savetxt("Q_matrix.csv",Q,delimiter=";",newline="\n")
        numpy.savetxt("W_matrix.csv",W,delimiter=";",newline="\n")
        numpy.savetxt("maximos.csv",maximos,delimiter=";",newline="\n")
        numpy.savetxt("minimos.csv",minimos,delimiter=";",newline="\n")

numpy.savetxt("cotas_sup_caja.csv",cotas_sup,delimiter=";",newline="\n")

numpy.savetxt("cotas_inf_caja.csv",cotas_inf,delimiter=";",newline="\n")
        f=open("resultados.txt","w")
        f.write("Generacion de caja en modelo compact core\n")
        f.write("\n")
        f.write("Reacciones eliminadas del modelo: ")
        f.write(str(a_elim))
        f.write("\n")
        f.write("Cantidad de flujos en el modelo: ")
        f.write(str(S.shape[1]))
```

```
    f.write("\n")
    f.write("Rango de la matriz estequiometrica resultante: ")
    f.write(str(rango))
    f.write("\n")
    if rango==S.shape[0]:
        f.write("El rango esta completo\n")
    else:
        f.write("ATENCION: el rango NO esta completo\n")
    f.write("Dimension del espacio de soluciones: ")
    f.write(str(dimension))
    f.write("\n")
    f.write("Volumen de la caja: {0:0.4e}\n".format(vol_caja))
    f.close()
os.chdir(current)
```

Anexo 12: Monte Carlo directo para cálculo del centro de masa en modelo E.coli core

```
import random
import time
import numpy

#MONTE CARLO UNIFORME
#Calculo del centro de masa

def centro(n,semilla):
    #Importar archivos
    W=numpy.loadtxt(open("W_matrix.csv","rb"),delimiter=";",skiprows=0)
    maximos=numpy.loadtxt(open("maximos.csv","rb"),delimiter=";",skiprows=0)
    minimos=numpy.loadtxt(open("minimos.csv","rb"),delimiter=";",skiprows=0)

cotas_sup=numpy.loadtxt(open("cotas_sup_caja.csv","rb"),delimiter=";",skiprow
s=0)

cotas_inf=numpy.loadtxt(open("cotas_inf_caja.csv","rb"),delimiter=";",skiprow
s=0)

    #Calcular volumen de la caja y dimensionalidad del espacio
    vol_caja=numpy.prod(numpy.array(cotas_sup)-numpy.array(cotas_inf))
    dimension=W.shape[1]

    #Inicializar
    random.seed(semilla)
    contar_adentro=0
    est_vol=0
    suma_cuadrados=0
    suma_centro_x=[]
    for i in range (0,dimension):
        suma_centro_x.append(0)
    suma_centro_x=numpy.array(suma_centro_x)
    start_time=time.time()

    #Generar archivo de salida
    auxnombretexto=["result_uniforme_seed",str(semilla),".txt"]
    nombretexto="".join(auxnombretexto)
    f=open(nombretexto,"w")

    f.write('Simulacion con distribucion UNIFORME en modelo compact core\n')
    f.write('VARIANDO LB DE VBM\n')
    f.write('CALCULO DEL CENTRO DE MASA\n')
    f.write('Hay %d iteraciones programadas\n'%n)
    f.write('Semilla = %d\n'%semilla)
    f.write("Cantidad de dimensiones del espacio de soluciones =
%d\n"%dimension)
    f.write('Volumen de la caja = {0:0.4e}\n'.format(vol_caja))
    f.write('-------------------------------------------\n')

    #Monte Carlo
    for j in range (1,n+1):
        xrand=[]
        for i in range (0,dimension):
            varx=random.uniform(cotas_inf[i],cotas_sup[i])
            xrand.append(varx)
```

```
        flujoscurrent=numpy.dot(W,xrand)
        if    all    (flujoscurrent[k]<=maximos[k]    for    k    in    range
(0,maximos.shape[0])):
            if    all    (flujoscurrent[k]>=minimos[k]    for    k    in    range
(0,minimos.shape[0])):
                contar_adentro=contar_adentro+1
                est_vol=est_vol+1
                suma_cuadrados=suma_cuadrados+1
                suma_centro_x=suma_centro_x+numpy.array(xrand)

        if                                  j                                  in
[1e1,1e2,1e3,1e4,1e5,1e6,1e7,1e8,1e9,1e10,1e11,1e12,1e13,1e14,1e15]:
            end_time=time.time()
            f.write('Han transcurrido %d iteraciones\n'%j)
            f.write('Elapsed   time   =   {0:0.2f}   seconds\n'.format(end_time-
start_time))
            f.write('Cantidad de puntos adentro = %d\n'%contar_adentro)
            f.write('Estimador = {0:0.4e}\n'.format(est_vol/j))
            f.write('Estimador    de    desv    estandar    del    estimador    =
{0:0.4e}\n'.format(((suma_cuadrados/(j-1)-j*((est_vol/j)**2)/(j-1))/j)**0.5))
            f.write('Volumen    estimado    del    espacio    de    soluciones    =
{0:0.4e}\n'.format(vol_caja*est_vol/j))
            f.write('Estimador    de    desv    estandar    del    volumen    =
{0:0.4e}\n'.format((((suma_cuadrados/(j-1)-j*((est_vol/j)**2)/(j-
1))/j)**0.5)*vol_caja))
            f.write('-------------------------------------------\n')
            f.flush()

    auxnombretexto=["centro_masa_x_seed",str(semilla),"_iter",str(n),".csv"]
    nombretexto="".join(auxnombretexto)

numpy.savetxt(nombretexto,suma_centro_x/contar_adentro,delimiter=";",newline=
"\n")

    f.close()
```