UNIVERSIDAD POLITÉCNICA DE VALENCIA

DEPARTAMENTO DE MATEMÁTICA APLICADA

DESARROLLO DE MÉTODOS DE SIMULACIÓN APLICADOS A LA OPTIMIZACIÓN DE FUNCIONES OBJETIVO BIOLÓGICAS

Tesis de Doctorado:

Ramón Alexander Jaime Infante

Directores:

Pedro José Fernández de Córdoba Castellá
Daniel Gamerman
Julián Triana Dopico

Valencia, 2018
Insitut Universitari de Matemàtica Pura i Aplicada

Pensamiento

Pregúntate *si lo que estás haciendo hoy te acerca al lugar en el que quieres estar mañana.*

Walt Disney

Dedicatoria

A mis padres, por ser el sosten en los momentos más dificiles.

Agradecimientos

Le agradezco especialmente a Pedro Fernandez de Cordoba, por su apoyo y todos los esfuerzos que ha realizados.

A Jualian Triana por su ayuda y su incondicional amistad.

A Daniel Gamerman por ayudarme tanto y exigirme más para ser mejor.

Resumen

La Biología de Sistemas es un campo de la investigación en el que confluyen varias disciplinas de conocimiento como la Física, Matemática, Química y Biología, donde las interacciones de los elementos internos de un microorganismo y el medio ambiente influyen en el desarrollo de procesos que se representan mediante un modelo matemático. Este enfoque permite comprender el funcionamiento de los sistemas biológicos y profundizar en el entendimiento de cómo sus interacciones conllevan a la aparición de nuevas propiedades y procesos.

En el estudio de los procesos biológicos, se realiza la confirmación o refutación de una teoría que se confronta con resultados experimentales. La Biología de Sistemas utiliza una hipótesis basada en el estudio de los procesos mediante una modelización matemática de los mismos.

Uno de los elementos principales de análisis en Biología de Sistemas es la reconstrucción de modelos metabólicos determinante a la hora de poder modificar el funcionamiento de un organismo determinado.

Este trabajo se aborda la automatización de esta actividad, así como los fundamentos esenciales de la Herramienta COPABI, como paso fundamental para una buena reconstrucción antes de aplicar diferentes métodos de optimización a un modelo metabólico a escala genómica.

Esta investigación se basa en métodos no tradicionales que permiten ofrecer mejoras en los resultados de las simulaciones, con un mejor acercamiento a la realidad en el contexto de la ingeniería metabólica.

Presentando **PyNetMet**, una librería de Python, como herramienta para trabajar con redes y modelos metabólicos. Con el fin de ilustrar las características más importantes

y algunos de sus usos, se muestran resultados de la herramienta como el cálculo de la agrupación media de las redes que representan a cada uno de los modelos metabólicos, el número de metabolitos desconectados en cada modelo y la distancia media entre dos metabolitos cualesquiera de la red.

Analizar los modelos metabólicos partiendo de la optimización monobjetivo no siempre se acerca todo lo deseado a la realidad, puesto que uno o más objetivos pueden entrar en conflicto porque tienen como denominador común la necesidad de elegir entre diferentes alternativas que han de evaluarse en base a diversos criterios. Para ello, se presentó un algoritmo de optimización multiobjetivo basado en algoritmos evolutivos que consiste en una adaptación del algoritmo sp-MODE implementado en la herramienta bioinformática **BioMOE**, que considera de manera simultánea la optimización de dos o más objetivos, a menudo en conflicto, dando como soluciones diferentes distribuciones de flujo en la que una no es mejor que la otra.

En el área de la comparación de modelos metabólicos se muestra una herramienta bioinformática llamada CompNet, basada en conceptos de teoría de grafos como las Redes de Petri, para poder establecer una comparación entre modelos metabólicos, determinando qué cambios serían necesarios para modificar determinadas funciones en uno de los modelos con respecto al otro, a través de la métrica Distancia de Edición.

Mediante las métricas de Baláž y Bunke se muestra el grado de semejanza que existe entre dos modelos mediante un valor cuantitativo que indica las semejanzas y diferencias ellos.

Abstract

Systems Biology is a field of research in which several disciplines of knowledge converge such as Physics, Mathematics, Chemistry and Biology, where the interactions of the internal elements of a microorganism and the environment influence the development of processes that are represented by a mathematical model. This approach allows us to understand how biological systems work and to deepen our understanding of how their interactions lead to the emergence of new properties and processes.

In the study of biological processes, the confirmation or refutation of a theory that is confronted with experimental results is performed. Systems Biology uses a hypothesis based on the study of processes by means of a mathematical modeling of them.

One of the main elements of analysis in Systems Biology is the reconstruction of metabolic models, which is decisive when it comes to modifying the functioning of a given organism.

This work addresses the automation of this activity, as well as the essential fundamentals of the COPABI Tool, as a fundamental step for a good reconstruction before applying different optimization methods to a metabolic model at genomic scale.

This research is based on non-traditional methods that allow us to offer improvements in simulation results, with a better approach to reality in the context of metabolic engineering.

Introducing PyNetMet, a Python library, as a tool for working with metabolic networks and models. In order to illustrate the most important characteristics and some of its uses, results of the tool are shown, such as the calculation of the mean grouping of the networks representing each of the metabolic models, the number of metabolites disconnected in each model and the mean distance between any two metabolites in the network.

Analyzing metabolic models on the basis of monobjective optimization does not always bring the desired closer to reality, since one or more objectives may come into conflict

because their common denominator is the need to choose between different alternatives to be evaluated on the basis of different criteria. To this end, a multi-target optimization algorithm based on evolutionary algorithms was presented, consisting of an adaptation of the sp-MODE algorithm implemented in the bioinformatics tool BioMOE, which simultaneously considers the optimization of two or more objectives, often in conflict, giving as solutions different flow distributions in which one is not better than the other.

In the area of the comparison of metabolic models, a bioinformatics tool called Network-Compare is shown, based on concepts of graph theory such as Petri dishes, in order to establish a comparison between metabolic models, determining what changes would be necessary to modify certain functions in one of the models with respect to the other, through the Editing Distance metric.

By means of the Baláž and Bunke metrics, the degree of similarity between two models is shown by means of a quantitative value that indicates the similarities and differences between them.

Resum

La Biologia de Sistemes és un camp de la recerca en què conflueixen diverses disciplines de coneixement com la Física, Matemàtica, Química i Biologia, on les interaccions dels elements interns d'un microorganisme i el medi ambient influeixen en el desenvolupament de processos que es representen mitjançant un model matemàtic. Aquesta perspectiva permet entendre el funcionament dels sistemes biològics i aprofundir en la comprensió de com les seves interaccions generen noves propietats i processos.

En l'estudi dels processos biològics, es realitza la confirmació o refutació d'una teoria que es confronta amb resultats experimentals. La Biologia de Sistemes utilitza una hipòtesi basada en l'estudi dels processos mitjançant una modelització matemàtica dels mateixos.
Un dels elements principals d'anàlisi en Biologia de Sistemes és la reconstrucció de models metabòlics determinants a l'hora de poder modificar el funcionament d'un organisme determinat.

En aquest treball s'aborda l'automatització d'aquesta activitat, així com els fonaments essencials de l'Eina COPABI, com a pas fonamental per a una bona reconstrucció abans d'aplicar diferents mètodes d'optimització a un model metabòlic a escala genòmica.

Aquesta investigació es basa en mètodes no tradicionals que permeten oferir millores en els resultats de les simulacions, amb una millor aproximació a la realitat en el context de l'enginyeria metabòlica.

Es presenta PyNetMet, una llibreria de Python, com a eina per treballar amb xarxes i models metabòlics. Per tal d'il·lustrar les característiques més importants i alguns dels seus usos, es mostren resultats de l'eina com el càlcul de l'agrupació mitjana de les xarxes que representen a cada un dels models metabòlics, el nombre de metabòlits desconnectats en cada model i la distància mitjana entre dos metabòlits qualssevol de la xarxa.

Analitzar els models metabòlics partint de l'optimització mono-objectiu no sempre s'acosta tot el desitjat a la realitat, ja que un o més objectius poden entrar en conflicte perquè tenen com a denominador comú la necessitat de triar entre diferents alternatives que han d'avaluar-se sobre la base de diversos criteris. Per a això, es va presentar un algoritme d'optimització multi-objectiu basat en algoritmes evolutius que consisteix en una adaptació de l'algoritme sp-MODE implementat en l'eina bioinformàtica BioMOE, que considera de manera simultània l'optimització de dos o més objectius, sovint en conflicte, donant com solucions diferents distribucions de flux en la qual una no és millor que l'altra.

En l'àrea de la comparació de models metabòlics es mostra una eina bioinformàtica anomenada CompNet, basada en conceptes de teoria de grafs com les Xarxes de Petri, per poder establir una comparació entre models metabòlics, determinant quins canvis serien necessaris per a modificar determinades funcions en un dels models respecte a l'altre, a través de la mètrica Distància d'Edició. Mitjançant les mètriques de Balaz i Bunke es mostra el grau de semblança que hi ha entre dos models a través d'un valor quantitatiu que indica les semblances i diferències entre ells.

Prefacio

Esta tesis doctoral fue desarrollada dentro del marco de colaboración entre el Grupo de modelización Interdisciplinar, InterTech (www.intertech.upv.es) de la Universidad Politécnica de Valencia y la Universidad de Pinar del Río. Lazos que se han mantenido por más de 15 años.

Las contribuciones originales del presente trabajo de investigación se recogen en los siguientes trabajos son:

- Gamermann D., Montagud A., Jaime Infante R. A., Triana J., Fernández de Córdoba P. and Urchueguía J. F. (2014) PyNetMet: Python tools for efficient work with networks and metabolic models. Computational and Mathematical Biology, 3(5): 1-11.
- Jaime-Infante R. A., Hernández-Martínez Z., Triana-Dopico J., Fosado-Tellez O., Montagud-Aquino A., Gamermann D., Fernández de Córdoba-Castellá P., Urchueguía-Schölzel J. F. (2014) Herramienta para la optimización de flujos metabólicos en sistemas biológicos. Investigación Operacional, 35(2): 96-103.
- Reyes R., Garrido J., Jaime R.A., Córdova V., Triana J., Villar L., Castro J.C.,Fernández de Córdoba P., Urchueguía J.F., Navarro E. y Montagud A.. 2011. Desarrollo de una plataforma computacional para el modelado metabólico de microorganismos. Nereis. Revista Iberoamericana de Métodos, Modelización y Simulación Interdisciplinar. Universidad Católica de Valencia "San Vicente Mártir". 3: 25-31. ISSN 1888-8550. 2011.

Actas de Congreso.

- XVII Convención y Feria Internacional Informática 2018. HERRAMIENTA DE ANÁLISIS MULTIOBJETIVO EN MODELOS METABÓLICOS APLICANDO ALGORITMOS EVOLUTIVOS. Ramón Alexander Jaime Infante, Maria Siurana Paula, Daniel Gamermann, Raymari Reyes Chirino. ISBN- 978-959-7255-00-0.
- XVII Convención y Feria Internacional Informática 2018. HERRAMIENTA PARA LA COMPARACIÓN DE MODELOS METABÓLICOS UTILIZANDO REDES DE PETRI. Ramón Alexander Jaime Infante, Raidel Rodríguez Romeu, Pedro José Fernández De Córdoba Castellá, Raymari Reyes Chirino. ISBN- 978-959-7255-00-0.
- II Congreso Internacional de Marketing, Desarrollo Local y Turismo (MARDELTUR 2017). Empleo de las TIC´s en el aprovechamiento de la biomasa forestal en Pinar del Río. Ramón Alexander Jaime Infante, Maria Siurana Paula, Daniel Gamermann, aidel Rodríguez Romeu, Pedro José Fernández De Córdoba Castellá. ISBN- 978-959-1634-50-4.

- XIV Convención y Feria Internacional Informática 2016. XI Congreso Internacional de Informática en la Salud 2016. La Habana (Cuba). “PYNETMET: PYTHON HERRAMIENTASPARA EL TRABAJO EFICIENTE CON LAS REDES Y MODELOS METABOLICOS”. Ramón Jaime Infante, Daniel Gamermann, Arnau Montagud Aquin, Julián Triana Dopico, Pedro Fernández de Córdoba-Castellá, Javier Fermín Urchueguía Schölzel. ISBN: 978-959-289-122-7.
- XIV Convención y Feria Internacional Informática 2016. XI Congreso Internacional de Informática en la Salud 2016. La Habana (Cuba), “Plataforma matemático-computacional para el análisis de Redes Metabólicas a Escala Genómica”. Julián Triana Dopico, Raidel Rodríguez Romeu, Zenén Hernández Martínez, Ramón Jaime Infante, Osvaldo Fosado Tellez, Arnau Montagud Aquin, Daniel Gamermann, Pedro Fernández de Córdoba-Castellá, Javier Fermín Urchueguía Schölzel. ISBN: 978-959-289-122-7.
- XII Jornada de Bioinformática. Sevilla, España, 2014. COPABI: a Computational Platform for Automationon the Genome Scale Metabolic Models Reconstruction R. REYES, M. SIURANA, D. GAMERMANN, A. MONTAGUD, J. TRIANA, R. JAIME, V.M. NINA, D. FUENTE, Y. PACHECO, J.F. URCHUEGUÍA, P. FERNÁNDEZ DE CÓRDOBA.
- Osvaldo Fosado Tellez, Ramón Alexander Jaime-Infante, Zenén Hernández Martínez, Julián Triana-Dopico, Raidel Rodríguez Romeu, Arnau Montagud Aquino, Daniel Gamermann, Pedro Fernández de-Córdoba-Castellá, Javier Fermín Urchueguía-Schölzel. 11th INTERNATIONAL CONFERENCE ON PERATIONS RESEARCH La Habana, March 11th-14th, 2014.
- R. Reyes, R. Jaime, J. Garrido, J. Triana, L. Villar, V. Córdova, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba, J.F. Urchueguía y J. Martínez. 2010. Diseño de bases de datos biológicas, un paso hacia la automatización del proceso de construcción de modelos a escala genómica. XV Convención Científica de Ingeniería y Arquitectura (CCIA 15). La Habana, Cuba, del 29 de noviembre al 3 de diciembre de 2010.Publicado en: Memorias de la Conferencia. ISBN 978-959-261-317-1.
- J. Garrido, J. Triana, L. Villar, R. Jaime, R. Reyes, V. Córdova, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba y J.F. Urchueguía. 2010. RationalOrganism Network Painter: una herramienta optimizada de visualización de redes metabólicas de fácil uso. XV Convención Científica de Ingeniería y Arquitectura (CCIA 15). La Habana, Cuba, del 29 de noviembre al 3 de diciembre de 2010. Publicado en: Memorias de la Conferencia. ISBN 978-959-261-317-1.
- J. Triana, V. Córdova, R. Jaime, R. Reyes, J. Garrido, L. Villar, F. Márquez, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba y J.F. Urchueguía. 2010. Modelo

metabólico de una cianobacteria, una fuente de energía a partir de la luz. I Congreso Internacional de Ingeniería Química, Biotecnológica y Alimentaria (CIIQBA 2010). La Habana, Cuba, del 29 de noviembre al 3 de diciembre de 2010. Publicado en: Memorias de la Conferencia. ISBN 978-959-261-317-1.

- J. Garrido, L. Villar, R. Reyes, R. Jaime, J. Triana, V. Córdova, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba, J.F. Urchueguía y J. Martínez. 2011. HYDRA: una plataforma informática orientada al diseño, análisis y visualización de redes metabólicas. XIV Convención y Feria Internacional Informática 2011. La Habana, Cuba, del 7 al 11 de febrero de 2011. Publicado en: Programa Científico. Pág. 55.ISBN 978-959-7213-01-7.
- R. Jaime, J. Garrido, R. Reyes, L. Villar, J. Triana, V. Córdova, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba, J. Urchueguía, J. Martínez y Z. Hernández. 2011. Nueva herramienta para el análisis del balance de flujo de las rutas metabólicas y su integración en Ron Painter. XIV Convención y Feria Internacional Informática 2011. La Habana, Cuba, del 7 al 11 de febrero de 2011. Publicado en: Programa Científico. Pág. 153. ISBN 978-959-7213- 01-7.
- R. Reyes, R. Jaime, J. Garrido, J. Triana, L. Villar, V. Córdova, J.C. Castro, E. Navarro, A. Montagud, P. Fernández de Córdoba, J.F. Urchueguía y J. Martínez. 2011. Base de datos biológica orientada a la automatización del proceso de construcción de modelos a escala genómica. Resultados en la Synechocystis SP PCC6803. XIV Convención y Feria Internacional Informática 2011. La Habana, Cuba, del 7 al 11 de febrero de 2011.Publicado en: Programa Científico. Pág. 215. ISBN 978-959-7213-01-7.

Contenido

Índice de Figuras

Índice de tablas

Capítulo 1. Introducción

1.1 Introducción

La Biología de Sistemas es un área de investigación que estudia los procesos de la vida desde un enfoque sistémico. Este campo requiere la colaboración de investigadores de diversos orígenes, incluyendo la Biología, Bioquímica, Informática, Matemáticas, Estadística, Física y Química. Estas colaboraciones son indispensables debido a la necesidad de integración a gran escala del conocimiento requerido para entender un determinado fenómeno biológico. Aquí, las fronteras entre las diferentes disciplinas desaparecen; con la creación de una nueva ciencia, que posee un novedoso universo de conocimientos, se abren caminos para la investigación donde se aplican métodos para un mejor análisis y predicción de situaciones. Esta tesis versa sobre un grupo de herramientas informáticas que se utilizarán para el análisis y optimización de modelos metabólicos, por lo que esta introducción, y una visión general de la Biología de Sistemas, tienen como objetivo entender mejor algunos conceptos de la Biología de Sistemas y conocer algunas de las herramientas que ya existen.

1.2 Biología de Sistemas

A principios de 1960, la biología experimentó acontecimientos trascendentales. La biología molecular había irrumpido, proporcionando una comprensión cada vez mayor del ADN, las proteínas y otros componentes químicos de las células. Una ciencia emergente, de gran rigor, que satisfacía las más altas exigencias de la química analítica y de la experimentación controlada y que destacaba la relevancia de las funciones bioquímicas y genéticas de las células y, de vez en cuando, de sus fenotipos. Así mismo, llevar a cabo aproximaciones holísticas en biología hace mucho más difícil el planteamiento de hipótesis, la realización de experimentos controlados o la ejecución del proceso científico para determinar el comportamiento de los sistemas y de las redes biológicas (Palsson *et al*., 2006).

La Biología de Sistemas (BS) es un campo de la investigación en el que confluyen varias disciplinas de conocimiento, donde las interacciones de los elementos internos de un microorganismo y el medio ambiente influyen en el desarrollo de procesos que se representan mediante un modelo matemático. Este enfoque permite comprender el funcionamiento de los sistemas biológicos y profundizar en el entendimiento de cómo sus interacciones conllevan la aparición de nuevas propiedades y/o procesos (Bard *et al*., 2013).

En el estudio de los procesos biológicos, donde se utilizan métodos científicos clásicos, se realiza la confirmación o refutación de una teoría que se confronta con resultados experimentales. La BS utiliza una hipótesis basada en el estudio de los procesos mediante una modelización matemática de los mismos. Como resultado de la simulación ejecutada a partir de los modelos matemáticos que se obtienen de la representación de los procesos, se obtiene una serie de predicciones del estado de dichos procesos biológicos que correspondería a los resultados experimentales esperados (Klipp *et al*., 2005).

Los sistemas biológicos pueden ser representados mediante un modelo metabólico, del cual se puede extraer la información para formar un sistema matemático representando la red que genera las interacciones entre los elementos que componen los procesos biológicos (Snoep *et al*., 2006).

Los valores de las interacciones entre dichos elementos a distintos tiempos y bajo diversas condiciones experimentales (simuladas), son predecibles debido a que la dinámica del estado de ese sistema modelado es matemáticamente calculable. Como se ha comentado con anterioridad, la BS es un área intrínsecamente interdisciplinar en la que participan informáticos, biólogos, químicos, matemáticos o físicos, entre otros.

La BS emplea diferentes aproximaciones a aquellas empíricas tradicionales, a través del estudio de sistemas biológicos en sus diferentes niveles, desde células y redes celulares hasta organismos completos. La BS involucra el reconocimiento de rutas metabólicas, de la interacción de proteínas y genes, así como el de los circuitos de organismos a nivel celular y de organismo completo, todo ello integrado en un modelo matemático.

Las principales características de la BS son (López *et al*., 2007):

- Estudia los sistemas biológicos a nivel molecular de una forma global.
- Contrasta con la aproximación clásica lineal (un gen, una proteína).
- Integra el conocimiento de diferentes plataformas o disciplinas (genómica, transcriptómica, proteómica, metabolómica, fisiológica, patológica, etc.).
- Maneja una gran colección de datos procedentes de estudios experimentales.
- Propone modelos matemáticos que pueden explicar algunos de los fenómenos biológicos estudiados.
- Proporciona soluciones matemáticas que permiten obtener predicciones para los procesos biológicos.
- Realiza estudios de comprobación de la calidad de los modelos descritos por medio de la comparación entre las simulaciones numéricas y los datos experimentales.

1.3 Modelos Metabólicos

Uno de los objetivos fundamentales de estudio en BS es la reconstrucción de redes metabólicas a escala genómica. Este proceso, no automatizado e iterativo, presupone el trabajo a largo plazo de un especialista utilizando la información contenida en diversas bases de datos biológicas, con el objetivo de organizar la lista de reacciones metabólicas específicas para un organismo (Förster *et al*., 2003).

Los modelos metabólicos a escala genómica se han desarrollado para predecir los fenotipos a nivel de sistemas, y han sido bien acogidos en los últimos años (O'Brien *et al.,* 2013). Muchos de ellos se han ampliado para incluir diferentes clases de información, por ejemplo, los niveles de expresión de productos génicos como las proteínas (Montagud *et al.*, 2010; O'Brien *et al*., 2013.), así como integrar muchos más procesos celulares y se pueden utilizar para simular estados celulares dinámicos (Karr *et al*. 2012). Así, los modelos metabólicos han sido útiles para la generación de nuevas hipótesis y la focalización de áreas promisorias dentro de la biotecnología (Esvelt y Wang, *et al.,* 2013).

El gran número de proyectos genómicos en curso, así como su alcance, indican la necesidad de herramientas totalmente automatizadas para generar nuevo conocimiento; los análisis computacionales predicen comportamientos de tipo biológico que sirven para predecir información biológica que, en su mayoría, se obtenía a través de exhaustivos esfuerzos de curación en los modelos metabólicos.

1.4 Red de Petri

Para representar gráficamente un modelo metabólico utilizamos una Red de Petri, que se describe como una herramienta de naturaleza gráfica para el diseño y análisis de sistemas dinámicos de eventos discretos (Murata *et al. 1989)*. Una red de Petri se representa gráficamente por un grafo dirigido bipartito.

Los lugares son dibujados como círculos y se usan para representar condiciones, las transiciones son dibujadas como barras y representan eventos. A cada arco se Ie asocia un peso estrictamente positivo. Por defecto, tal peso tiene un valor de uno.

En una Red de Petri también existe la Marcación, que no es más que un conjunto de dibujos en forma de pequeños puntos dentro de los lugares (círculos) y que se utilizan para mostrar la evolución de la red. Su número y posición cambian durante su ejecución (Silva *et al. 1985)*.

Los lugares y transiciones se conectan por arcos dirigidos. Un arco dirigido de un lugar P_i a una transición T_j define un lugar de entrada de la transición. Múltiples entradas a una transición son indicadas por múltiples arcos desde el lugar de entrada a la transición. Un lugar de salida es indicado por un arco desde la transición al lugar. Análogamente, múltiples salidas son representadas por múltiples arcos.

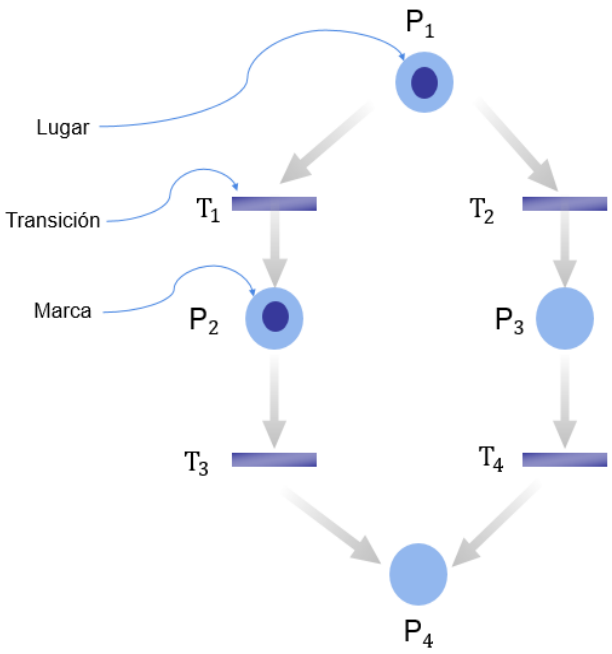


Figura 1: Lugares, transiciones y marcas en una Red de Petri.

.El marcado de una Red se define por un vector columna en donde los elementos son el número de marcas contenidas en los lugares. El marcado define el estado de la Red de Petri. Para la Red de Petri dada en la figura 1.1 tenemos: Lugares: $P = \{P_1, P_2, P_3, P_4\}$; Transiciones: $T = \{T_1, T_2, T_3, T_4\}$; Marcado $m_1 = m_2 = 1$; $m_3 = m_4 = 0$; $M = [1\ 1\ 0\ 0]$ (Silva *et al. 1985)*.

En nuestro caso podemos decir que los metabolitos son los lugares o vértices, las reacciones serían las transiciones y los arcos que conectan los lugares con las transiciones son las aristas del grafo donde el peso de estos arcos sería el coeficiente estequiométrico de la reacción.

De la misma manera que analizamos la Red de Petri también tenemos que introducir algunos conceptos sobre los grafos como el de igualdad de grafos, porque la Red de Petri es un caso específico de grafo.

1.5 Métodos de análisis en Biología de Sistemas

Hay tres tipos de modelos metabólicos (Baart y Martens, *et al.,* 2012):

a) Los modelos de estado estable, que sólo tienen en cuenta el flujo de metabolitos a través del sistema en estado estacionario.

b) Los modelos cinéticos en estado de equilibrio, que tienen en cuenta el flujo de metabolitos a través del sistema en estado estacionario y que contienen al menos una ecuación cinética que relaciona la concentración de un metabolito a una velocidad de reacción.

c) Los modelos cinéticos dinámicos, que tienen en cuenta la cinética de todas las diferentes enzimas que participan en la red de reacciones.

En una situación ideal, el modelo contiene diferentes niveles de información, incluyendo estequiometría de reacción y cinética de reacción así como elementos de regulación metabólica. Sin embargo, la información cinética y de regulación requerida es muy escasa, a pesar de que hay varios estudios en esta dirección (Tomita *et*, 2001; Covert *et al.*, 2004; Bruggeman y Westerhoff *et al.*, 2006; Bruggeman *et al*, 2008; De Mey *et al*., 2010). Por esta razón, los investigadores generalmente utilizan una aproximación al estado estacionario, utilizando la estequiometría de la reacción en combinación con el balance de masas de los flujos metabólicos (Taymaz-Nikerel *et al*., 2010). Las capacidades metabólicas de la red se pueden calcular usando métodos de simulación por ordenador basados en restricciones. Para este fin están disponibles una gran variedad de herramientas y algoritmos y pueden ser aplicados en los modelos metabólicos. Entre ellos, podemos encontrar el Análisis del Balance de Flujos (FBA) (Varma y Palsson *et al*., 1993; Edwards *et al*, 1999), Análisis de Flujos Metabólicos (AMF) (Schilling *et al.*, 1999; Varma y Palsson *et al*., 1994c), Primaria Flux Modos (EFM) (Schuster *et al*., 1999), Análisis de Rutas Extremas (EXPA) (Schilling *et al*., 1999), Análisis de Robustez (RNA) (Edwards y Palsson *et al*., 2000), Análisis de Planos de Fase Fenotípicas (Phpp) (Edwards *et al*., 2002), Minimización de Ajuste Metabólico (MOMA) (Segrè *et al*., 2002), Análisis de la Variabilidad de Flujos (FVA) (Mahadevan y Schilling *et al*., 2003) y Normativa On-Off Minimización (ROOM) (Shlomi *et al*., 2005). Estos métodos, entre otros, se pueden clasificar, de acuerdo a su finalidad, en categorías (véase la Figura 2.1), tales como: la búsqueda de mejores estados óptimos en un rango admisible; el estudio de los posibles estados permitidos; alterar posibles fenotipos como consecuencia de variaciones genéticas; y definir e imponer restricciones adicionales (Price *et al*., 2004b).

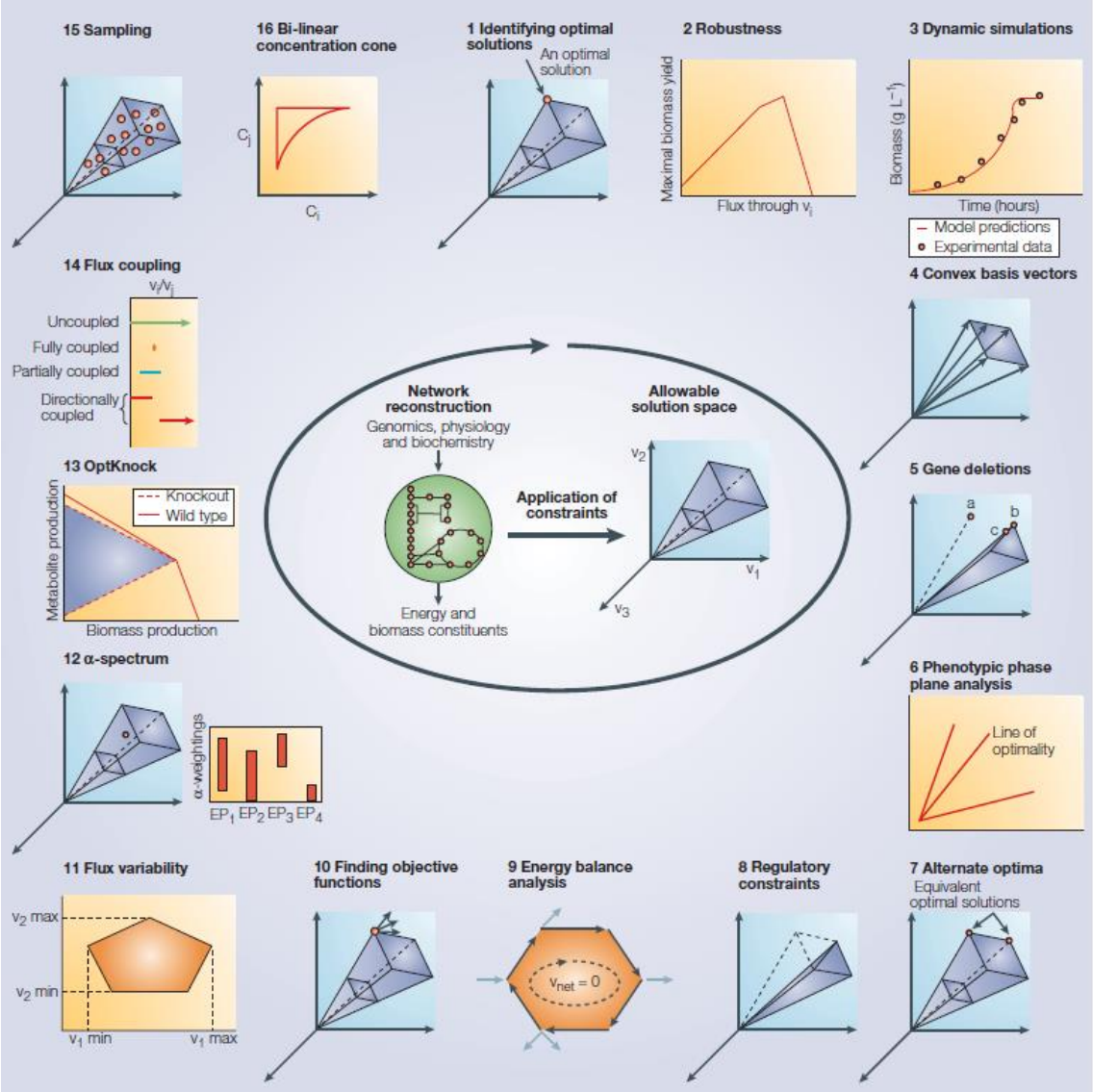


Figura 2: Resumen de algunas herramientas para el análisis basado en restricciones. Tomado de (Price et al., 2004b).

1.6 Las restricciones en las funciones celulares

El proceso de evolución está intrínsecamente asociado a las ciencias biológicas. Los sistemas biológicos se han adaptado a diversos ambientes de la historia evolutiva, y conforme se van replicando, producen descendientes que no son genéticamente idénticos a los padres, por lo tanto la generación de una población tiene ligeras diferencias entre los individuos. Como Palsson escribió en 2006: "Para sobrevivir en un entorno determinado, los organismos deben satisfacer innumerables restricciones, lo que limita el rango de fenotipos disponibles. Todos los fenotipos expresados resultantes del proceso de selección deben satisfacer un conjunto de restricciones. Por tanto, la identificación de las restricciones para definir rangos de estados fenotípicos permisibles

proporciona un enfoque fundamental para la comprensión de los sistemas biológicos consistente con nuestra comprensión de la forma en que el organismo funciona y evoluciona" (Palsson *et al*., 2006).

Los diferentes tipos de restricciones que limitan las funciones celulares tendrán que tener una presencia en nuestros modelos. Hay dos restricciones ajustables. La primera puede ser utilizada para restringir la gama de posibles comportamientos, por ejemplo, los factores físico-químicos, como la conectividad o la capacidad caracterizada por el rango de flujos máximo y mínimo; y la segunda se puede utilizar para limitar aún más los comportamientos permisibles, tales como: las velocidades de reacción o factor de capacidad ajustable por regulación transcripcional. Hay que tener en cuenta que estas limitaciones se pueden ajustar a través de un proceso evolutivo o mediante el cambio de las condiciones ambientales. Además, las limitaciones ajustables pueden variar en una población de un individuo a otro. Hay diversas formas de clasificar estas restricciones y muchos autores las han discutido desde diferentes puntos de vista. Algunos de estos autores (Price *et al.*, 2004b.) mencionan cuatro categorías de restricciones que son: restricciones físico-químicas, restricciones espaciales o topológicas, restricciones ambientales y restricciones reglamentarias o autoimpuestas.

a) Restricciones físico-químicas:

Muchas de estas restricciones rigen los procesos celulares, son inviolables y proporcionan restricciones "duras" en las funciones celulares. La conservación de la masa, la energía y el momento, representan restricciones duras. El interior de una célula está densamente empaquetado, formando un entorno donde la viscosidad puede ser del orden de 100-1000 veces la del agua. Por tanto, las velocidades de difusión dentro de una célula pueden ser lentas, especialmente para macromoléculas (Elowitz *et al*., 1999). El confinamiento de un gran número de moléculas dentro de una membrana semipermeable causa una alta osmolaridad. Por tanto, las células requieren mecanismos para hacer frente a la presión osmótica generada, tales como bombas de sodio-potasio o una pared celular (Werner y Heinrich *et al*., 1985; Lew y Bookchin *et al*., 1986). Las velocidades de reacción intracelulares están determinadas por las concentraciones locales dentro de las células y pueden estar limitadas por la masa y el

transporte de las moléculas. Las reacciones tienen velocidades de reacción máximas (denotados por Vmax) y algunas se estiman en aproximadamente un millón de moléculas por μm^3 por segundo. Por otra parte, las reacciones bioquímicas deben dar lugar a un cambio de energía libre negativa que permita el sentido de la reacción directa.

b) Restricciones topológicas:

La aglomeración de moléculas dentro de las células conduce a las restricciones topológicas o tridimensionales. Por ejemplo, la dimensión lineal del genoma bacteriano es del orden de 1.000 veces la longitud de la célula. Por tanto el ADN (ácido desoxirribonucleico) debe quedar focalizado en la región nuclear en una configuración accesible y funcional ya que el ADN es sólo funcional si es accesible.

c) Restricciones ambientales:

Estas restricciones sobre las células, tales como la disponibilidad de nutrientes, pH, temperatura, osmolaridad, la disponibilidad de receptores de electrones, etc., suelen ser temporales y condicionalmente dependientes. Las restricciones ambientales son de importancia fundamental para el análisis cuantitativo de las funciones de los microorganismos. Sin embargo, el medio ambiente natural puede ser difícil de definir con precisión. Se necesitan medios definidos y condiciones ambientales bien documentadas para integrar estos datos en modelos cuantitativos precisos, descriptivos y predictivos. A menudo, los experimentos de laboratorio con un medio de cultivo determinado son de uso limitado para el modelado *in silico*.

d) Restricciones regulatorias:

Estas restricciones son fundamentalmente diferentes de las tres categorías mencionadas anteriormente. Son auto-impuestas, están sujetas a cambios evolutivos y, por tanto, pueden variar en el tiempo. Por esta razón, estas restricciones pueden ser referidas como reguladoras o regulatorias, en contraste con las restricciones físico-químicas, las topológicas y ambientales que dependen del tiempo. Sobre la base de las condiciones ambientales, las restricciones regulatorias proporcionan un mecanismo para eliminar estados fenotípicos subóptimos y limitan las funciones celulares a los comportamientos de aumento de capacidades. Las restricciones regulatorias se

evidencian en una célula con una variedad de maneras, incluyendo la cantidad de productos génicos expresados (regulación transcripcional y traduccional) y su actividad (regulación de la enzima).

Las restricciones se pueden aplicar al análisis de redes biológicas reconstruidas, por ejemplo, en el cálculo de las velocidades de reacción en las redes metabólicas, a comportamientos alcanzables y se pueden aplicar de una manera sucesiva. La imposición de una restricción conduce a espacios de soluciones más allá que el cálculo de una única solución, son el sello distintivo de los modelos basados en la teoría. Los estados funcionales asociados a comportamientos celulares pueden alcanzarse en este espacio solución. Cada comportamiento permisible básicamente representa un fenotipo diferente, basado en la composición molecular, las propiedades bioquímicas de los componentes, su interconectividad y las restricciones impuestas. Los enfoques basados en restricciones determinan buena parte de los procedimientos de análisis *in silico* que son útiles en la interpretación, y en ocasiones, en la predicción de la relación genotipo-fenotipo.

1.7 Métodos de análisis basados en restricciones

Los procedimientos de análisis y de reconstrucción basados en restricciones (COBRA del inglés *Constraint based reconstruction and analysis*) para el análisis de los estados fenotípicos permisibles de microorganismos a escala genómica se han desarrollado rápidamente en los últimos años (Price *et al*., 2004; Reed *et al*., 2003). Una de las metodologías implementadas incluye tres pasos fundamentales (Palsson 2006):

1) Reconstrucción de la red metabólica a escala genómica.
2) Formulación de las restricciones apropiadas en la generación de los modelos matemáticos a optimizar.
3) Aplicación de varios métodos de análisis para evaluar las propiedades de modelos a escala genómica.

Es importante tener en cuenta que sólo una reconstrucción minuciosa y de calidad puede conducir al modelo a escala genómica de un organismo específico. Muchos

procedimientos automatizados generan redes de reacciones que no pueden ser utilizadas como base para el cálculo. Las razones para esto varían; a menudo, las reglas básicas de química suelen ser violadas y, por consiguiente, las vías metabólicas pueden tener incongruencias. La conversión desde una red a escala genómica a un modelo a escala genómica es un proceso laborioso y detallado, que se ha logrado para un grupo extenso de organismos. Para analizar estos modelos a escala genómica se han desarrollados muchos métodos de análisis que pueden ser clasificados en varias categorías (Palsson, 2006):

1. Métodos de análisis para encontrar el "mejor" estado óptimo dentro del rango permitido y el estudio de las propiedades de tales soluciones. Los métodos de esta categoría involucran los cálculos de una única solución óptima, y el estudio de soluciones alternativas o equivalentes. Adicionalmente, se puede estudiar la variabilidad de flujo entre todas las soluciones equivalentes.
2. Métodos de análisis para investigar cómo varían los valores de los flujos al cambiar uno, dos, o más parámetros, y para investigar cuán acopladas pueden estar las reacciones (correlacionadas) en todos los estados funcionales de una red metabólica. A menudo, la influencia de los parámetros ambientales puede variar en un rango numérico especificado.
3. Métodos para evaluar los posibles cambios fenotípicos como consecuencia de las variaciones genéticas asociadas a la eliminación de uno o más genes y, por tanto, de los productos génicos. Estos procedimientos se han desarrollado hacia el diseño de redes metabólicas a escala genómica.
4. Métodos de análisis para definir y establecer restricciones adicionales.

1.8 Métodos de optimización

Muchos de los métodos de análisis desarrollados se basan en el uso de diversas técnicas de optimización. Estos métodos incluyen (Cases *et al*., 2003; Bazaraa *et al*., 1993):

- La programación lineal (LP, del inglés “Linear Programming”): este método se utiliza cuando el problema a ser resuelto involucra un conjunto lineal de restricciones (igualdades y desigualdades) y una función objetivo lineal.

- Programación cuadrática (QP, del inglés “Quadratic Programming”): se utiliza este método cuando el problema a resolver consiste en un conjunto lineal de restricciones (igualdades y desigualdades) y una función objetivo cuadrática. Un objetivo cuadrático surge, por ejemplo, cuando se utiliza una distancia euclidiana como una función objetivo. Cuando se calcula la distancia euclidiana, los elementos de un vector son elevado a una segunda potencia. El "método de mínimos cuadrados" es un ejemplo de QP.
- Programación entera mixta lineal (MILP): la formulación de problemas de LP a menudo conduce a la utilización de variables discontinuas. En ocasiones, las variables lógicas se introducen y adquieren un valor de 0 ó 1. MILP se utiliza para resolver este tipo de problemas.
- Programación no lineal (NLP): los problemas de optimización más complicados implican el uso de restricciones no lineales y/o una función objetivo no lineal. En general, estos problemas son difíciles de resolver. Una cuestión fundamental que se plantea es que el espacio de soluciones es no convexo; en tal circunstancia, no se puede garantizar la existencia en el espacio del óptimo global para la función objetivo.

1.9 Herramientas informáticas

Con la utilización de los métodos de optimización se han desarrollado una gran variedad de técnicas para el análisis de redes metabólicas, incluyendo el modelado dinámico, que considera aspectos cinéticos asociados a las concentraciones temporales de los metabolitos (Teusink *et al*., 2000); análisis de control metabólico (MCA del inglés Metabolic Control Analysis), que evalúa los efectos de pequeñas variaciones de concentraciones de enzimas o metabolitos (Kell y Mendes 2000); y, más recientemente, el análisis de balance energético (EBA del inglés Energy Balance Analysis), que estudia una red al tiempo que aplica las leyes de termodinámica.

Toda expresión fenotípica de un sistema biológico dado debe satisfacer las restricciones básicas que se imponen a las funciones de todas las células (Covert *et al*., 2003). Como se mencionó anteriormente, estas restricciones son fisicoquímicas, topológicas y

ambientales. En pocas palabras, las limitaciones fisicoquímicas están relacionadas con leyes físicas como la conservación de la masa y la energía; las restricciones topológicas representan restricciones espaciales de los metabolitos dentro de compartimentos celulares; y las limitaciones ambientales incluyen disponibilidad de nutrientes, pH y temperatura que varía con el tiempo y el espacio. Estas restricciones pueden ser descritas matemáticamente, y el conjunto de las posibles soluciones a estas condiciones matemáticas proporciona la gama de estados válidos del sistema biológico. Estas restricciones estrechan el espectro de posibles fenotipos y proporcionan un enfoque más específico en función de la red celular (Price *et al*., 2004).

En tal sentido, el Análisis del Balance de Flujos (FBA, del inglés *"Flux Balance Analysis"*) se refiere específicamente a derivar un conjunto factible de flujos de estado estacionario que optimiza un objetivo celular indicado, por ejemplo, maximizando la producción de biomasa dentro de una red metabólica, sujeta a un conjunto de restricciones de conservación de masa (Price *et al*., 2004). Una vez que se identifica este conjunto de flujos en el estado estacionario, las técnicas de optimización pueden utilizarse para evaluar el rendimiento del sistema biológico ante diversas perturbaciones tales como: diferentes objetivos celulares o variabilidad en las restricciones impuestas (Segre *et al*., 2002). Los conjuntos de flujos resultantes pueden compararse entre sí y con los datos experimentales, y, en última instancia, la búsqueda de posibles flujos puede producir modelos predictivos de redes bioquímicas a gran escala expuestas a diferentes condiciones.

Las opciones más comunes de funciones objetivos usadas en el FBA incluyen la producción de biomasa (Varma y Palsson, 1994; Edwards y Palsson 2000), la energía (Ramakrishna *et al*., 2001) o los metabolitos de interés industrial (Varma *et al*., 1993). El fundamento de este análisis se basa en que el organismo ha evolucionado para maximizar o minimizar su rendimiento bajo condiciones a las que se adaptó anteriormente. Las modificaciones genéticas a través de mutaciones de genes no se pueden describir con precisión con los objetivos utilizados para las células de tipo salvaje. Con el fin de comprender mejor los flujos de los estados mutantes, se propuso una reformulación del FBA denominada MOMA (del inglés, *Minimization of Metabolic Adjustment*) (Segrè *et al*., 2002). Mediante la programación cuadrática, MOMA calcula

los flujos metabólicos de los mutantes a través de la minimización de la distancia euclidiana entre dos espacios vectoriales, el correspondiente a la célula tipo salvaje y al del estado mutado. Mientras que la selección natural es una razón válida para esta hipótesis, en el caso de organismos de tipo salvaje, se puede esperar que los organismos mutantes no hayan alcanzado su rendimiento óptimo hipotético. MOMA (Segrè *et al*., 2002) direcciona este problema mediante la prueba de hipótesis de que la deleción de un gen provoca una redistribución de flujos mínimos con respecto al metabolismo de tipo salvaje, de forma compatible con la ausencia de la reacción eliminada.

El Análisis de Balance de Flujo Dinámico (DFBA, del inglés *Dynamic Flux Balance Analysis*) es una extensión del FBA que tiene la ventaja de permitir el análisis de las interacciones entre el metabolismo y el medio ambiente. Este enfoque genera predicciones dinámicas de las variaciones temporales del sustrato y la biomasa (Hjersted y Henson, 2009). DFBA ofrece un modelo estructurado del proceso bioquímico donde las reacciones metabólicas cambian dependiendo de las condiciones ambientales, que se representa por los cambios en la dependencia funcional de los sustratos.

Por otra parte, predecir el estado metabólico del organismo después de mutaciones génicas es una tarea difícil, ya que el sistema de regulación rige una serie de cambios metabólicos transitorios que convergen a una condición de estado estacionario. Otra de las herramientas que se fundamenta sobre un algoritmo basado en restricciones es Regulatory on_off minimization (ROOM), que permite la predicción del estado de equilibrio metabólico después de la introducción de mutaciones. Su objetivo es minimizar significativamente el número de cambios de flujo con respecto al tipo salvaje. ROOM predice con precisión los flujos metabólicos en estado estacionario que mantienen la linealidad de flujo, de acuerdo con las medidas de flujo experimentales, e identifica correctamente las vías alternativas cortas utilizadas para reencaminar el flujo metabólico en respuesta a las mutaciones (Shlomi *et al*., 2005).

Capítulo 2. Automatización en la reconstrucción de modelos metabólicos

2.1 Introducción

La reconstrucción de modelos metabólicos a escala genómica es determinante a la hora de poder modificar el funcionamiento de un organismo determinado. Es una tarea muy compleja y cuando se realiza de manera manual puede llevar mucho tiempo. La automatización de esta actividad, así como los fundamentos esenciales de la Herramienta COPABI, los tratamos en este capítulo pues es fundamental una buena reconstrucción como paso previo antes de aplicar diferentes métodos de optimización a un modelo metabólico a escala genómica.

2.2 La reconstrucción de modelos metabólicos

Uno de los elementos principales de análisis en Biología de Sistemas es la reconstrucción de modelos metabólicos. Este proceso, cuando se realiza de manera artesanal, es engorroso, iterativo y extenso, convirtiéndose en el trabajo a largo plazo de un especialista utilizando la información contenida en diversas bases de datos biológicas, con el objetivo de organizar la lista de reacciones metabólicas específicas para un organismo (Förster *et al.*, 2003). En los últimos tiempos, los científicos han desarrollado y puesto a prueba diversos métodos y herramientas de gran utilidad en el desarrollo de dichas investigaciones. Con la reconstrucción adecuada de un modelo metabólico habríamos representado parcialmente un organismo virtual al cual le adicionaríamos los cambios en el metabolismo a través de perturbaciones ambientales y genéticas. Con las restricciones apropiadas y la simulación a través de análisis computacionales, se pueden explorar posibles distribuciones de flujos dentro de la célula en diferentes condiciones medioambientales y/o para una determinada configuración genética (Montagud *et al.*, 2010).

La reconstrucción de modelos metabólicos a escala genómica de organismos permite la integración de información genómica con actividades metabólicas observadas a través de experimentos fenotípicos y otras mediciones "ómicas" para obtener conocimiento biológico oculto y que pudiera ser de otro modo difícil de obtener (Fang *et al.*, 2011).

La reconstrucción consiste en reunir toda la información referente al metaboloma de una especie, además de tener los genes que codifican a las enzimas que catalizan cada una de las reacciones metabólicas. Algo más que hay que tener en cuenta son las coenzimas y cofactores necesarios para la catálisis enzimática, la estequiometría y reversibilidad de las reacciones, información de la composición de la biomasa y aspectos de la regulación metabólica (Förster *et al.*, 2003).

Se han desarrollado diversos proyectos en estos temas, los cuales han sido enfocados hacia la reconstrucción de modelos metabólicos de organismos con diferentes fines, como la producción optimizada de biocombustibles de tercera y cuarta generación mediante microorganismos como cianobacterias y levaduras (Montagud *et al.*, 2010; Triana *et al.*, 2014), la reconstrucción del modelo metabólico de *Burkholderia cenocepacia* J2315 para el estudio de tratamientos orientados a pacientes que padecen Fibrosis Quística (FQ) (Fang *et al.*, 2011), la capacidad de *Rhodobacter sphaeroides* para producir hidrógeno, polihidroxibutirato u otros hidrocarburos, que lo convierten en un excelente candidato para su uso en una amplia variedad de aplicaciones biotecnológicas (Imam *et al.*, 2011), el estudio de *Clostridium beijerinckii* como un organismo que posibilita mejorar la producción de butanol (Milne *et al.*, 2011) etc.

2.3 Herramientas informáticas orientadas a la reconstrucción de los modelos metabólicos

Existen varios paquetes informáticos que asisten en la automatización de la reconstrucción de los modelos metabólicos a escala genómica y se realizan grandes esfuerzos encaminados a reducir el tiempo que toma la reconstrucción de tales modelos. Es preciso destacar algunos ejemplos como el desarrollado por Peter Karp, publicado como Pathways Tools en varias versiones. Esta aplicación consta de varios módulos: PathoLogic, Pathway Hole Filler, Pathway Tools Navigator y el Editor Functions. Lo más importante de estas herramientas es que incluyen la utilización del genoma anotado de cualquier sistema biológico para inferir vías metabólicas probables existentes en el mismo, así como para generar una nueva base de datos de vías metabólicas y genomas (Karp *et al.*, 2002). Recientemente, este autor y colaboradores, han expuesto un método para el rellenado múltiple de huecos en la red metabólica

acelerando la generación de modelos estequiométricos. Llamada MetaFlux, la aplicación se basa en la utilización de la programación MILP (del inglés, *Mixed Integer Linear Programming*) para corregir el conjunto de reacciones, metabolitos relacionados con la ecuación de biomasa, nutrientes y procesos de secreción celular. El método genera los modelos metabólicos directamente de la base de datos de vías metabólicas y genomas gestionada por el Pathways Tools (Latendresse *et al.*, 2012).

Existen otros algoritmos implementados en diferentes herramientas como son la IdentiCS (del inglés, *Identification of Coding Sequences from Unfinished Genome Sequences*), metaSHARK (del inglés, *metabolic SearcH And Reconstruction Kit*) así como AUTOGRAPH (del inglés, *AUtomatic Transfer by Orthology of Gene Reaction Associations for Pathway Heuristics*). La aplicación IdentiCS consiste en el uso de secuencias genómicas no-anotadas para predecir regiones codificadoras y funciones asociadas a ellas, así como para la construcción *in silico* de la red metabólica (Sun and Zeng, 2004). El paquete informático metaSHARK se basa en la detección de genes codificadores de enzimas dentro del genoma no anotado de un sistema biológico y su visualización en el contexto de una red metabólica. Los dos módulos que lo componen SHARKhunt y SHARKview, ofrecen un nivel mejorado de flexibilidad y precisión en la automatización en la anotación de enzimas. El método implementado demuestra su utilidad en la generación de conocimiento en el metabolismo celular y por tanto en la reconstrucción de modelos metabólicos, a partir de genomas no anotados (Pinney *et al.*, 2005). El método publicado como AUTOGRAPH se auxilia en la disponibilidad de los modelos metabólicos bien curados y publicados en artículos científicos para predecir de forma eficiente una equivalencia genética entre especies. Esto permite la transferencia hacia el modelo metabólico del organismo en estudio de la asociación gen/reacción a partir de la red metabólica publicada (Notebaart *et al.*, 2006).

Otras aplicaciones en este campo son las basadas en tecnología web, algunas de ellas son ReMatch, Model SEED y FAME. El algoritmo propuesto en ReMatch hace coincidir modelos metabólicos desarrollados por usuarios con la información de una base de datos interna, que incluye un glosario de nombres de metabolitos, generada a partir de KEGG, MetaCyc y CheBI. Por otro lado, ReMatch es capaz de aumentar el número de reacciones en el modelo, incorporadas a partir de la base de datos interna o introducidas

por el usuario (Pitkänen *et al.*, 2008). El recurso vía web Model SEED pretende analizar, comparar, reconstruir y curar la red metabólica a escala de sistemas biológicos. En este particular, los usuarios pueden enviar las secuencias genómicas en el sistema de anotación RAST y como resultado se construye una red de reacciones metabólicas, establece una relación de asociación gen-proteína-reacción, así como la BM para cada genoma analizado (Henry *et al.*, 2010). Por otro lado, FAME es una herramienta de modelado que asiste en la creación, edición y análisis de modelos metabólicos de microorganismos a partir de la información de KEGG (Boele *et al.*, 2012).

Los servidores web constituyen otro recurso en la automatización de la reconstrucción de redes metabólicas. Cabe destacar algunos como MrBac (del inglés, *Metabolic network Reconstructions for Bacteria*), que intentan construir un borrador de la red metabólica. Esta herramienta integra análisis de genómica comparativa, recuperación de anotaciones del genoma, así como generación de archivos con formatos estándares en Biología de Sistemas (Liao *et al.*, 2011).

Las plataformas computacionales también ofrecen asistencia en esta labor. Ejemplos de estas son: MicrobesFlux y GEMSiRV (del inglés, GEnome-scale Metabolic model imulation, Reconstruction and Visualization). MicrobesFlux es una plataforma web capaz de descargar automáticamente la red metabólica de aproximadamente 1200 especies depositadas en KEGG y convertirlas en borradores de los modelos metabólicos correspondientes. Además, la plataforma también proporciona herramientas personalizadas que permiten delecionar genes e introducir vías metabólicas heterólogas (Feng *et al.*, 2012). Por su parte, GEMSiRV proporciona funcionalidades para la construcción y edición de redes metabólicas, para la visualización de redes integrando datos experimentales, así como herramientas de análisis, por ejemplo, FBA. Adicionalmente, todos los modelos metabólicos GEMSiRV generados incluyendo proyectos en progreso, pueden ser fácilmente intercambiados por la comunidad científica (Liao *et al.* 2012). Otro ejemplo lo constituye el trabajo realizado por el grupo de Modelización Interdisciplinar InterTech, de la Universidad Politécnica de Valencia, en el desarrollo de HYDRA (del inglés, Hybrid Draw and Routes Analysis), una plataforma computacional orientada al diseño, análisis y visualización de redes metabólicas (Garrido *et al.*, 2011; Reyes *et al.*, 2011).

2.4 ¿Por qué se construye COPABI?

Varios protocolos han sido publicados para definir en detalle cada uno de los pasos que deben dar lugar a una reconstrucción adecuada, así como los paquetes de software y bases de datos que pueden ayudar en esta labor (Förster *et al.*, 2003; Notebaart *et al.*, 2006; Feist *et al.*, 2009; Thiele y Palsson, 2010). En la actualidad, el proceso de reconstrucción es largo y arduo, principalmente debido a su construcción manual y por los controles de calidad (Thiele y Palsson, 2010). Algunos informes afirman que la reconstrucción de una red metabólica a escala genómica puede tomar fácilmente desde varios meses hasta 2 años (Förster *et al.*, 2003; Duarte *et al.*, 2007). Por otra parte, algunos trabajos han intentado automatizar la reconstrucción metabólica, o al menos algunas partes del mismo, con el fin de reducir el tiempo necesario para un proyecto de este tipo. Sin embargo, estos esfuerzos se han visto obstaculizados por los problemas actuales en las bases de datos que contienen la información y las anotaciones del genoma (Feist *et al.*, 2009). De este modo, los algoritmos resultantes todavía necesitan la supervisión de expertos con el fin de ser capaz de generar modelos de redes metabólicas de calidad como base para el análisis predictivo (Thiele y Palsson, 2010).

La gran variedad de herramientas informáticas y de algoritmos con los cuales han sido implementadas éstas corroboran los amplios esfuerzos que se realizan para automatizar el proceso de reconstrucción (Karp *et al*., 2002). A pesar de todo esto, los modelos generados con cada una de estas herramientas siempre necesitan de un proceso de refinamiento manual por parte de los especialistas, porque es cierto que estas herramientas aún mantienen limitaciones. Aun cuando muchos de estos métodos gestionan de manera precisa la información de las grandes Bases de Datos, se siguen generando listas de reacciones no asociadas a las vías metabólicas con las que se relaciona. Además, incluyen reacciones metabólicas asociadas a procesos genéticos moleculares, como es el caso de las reacciones de modificación y reparación del ADN, o a los procesos de replicación y transcripción del ADN o a los procesos de división celular, reacciones de transferencia de señales, etc. Todas estas reacciones forman parte del metabolismo celular pero no son útiles a la hora de construir los modelos metabólicos, por lo que no deben incluirse (Reyes, 2013).

De acuerdo con varios protocolos descritos en la literatura se requieren de 3 a 4 pasos fundamentales para la reconstrucción a escala genómica de modelos metabólicos (Thiele and Palsson, 2010; Triana *et al.*, 2014). Lo primero en este proceso es generar un borrador del modelo metabólico del organismo en que se desea reconstruir, seguido de un refinamiento exhaustivo (desde el principio de ese paso se debe tener en cuenta el formato computacional para la escritura del modelo), así como de una validación del mismo a través de las capacidades de la red metabólica. La curación iterativa del modelo que incluye el análisis de huecos en la red, así como el ajuste metabólico para su llenado, la eliminación de ciclos fútiles, entre otras, son parte crucial en el proceso de depurado. Se conocen varios métodos que inciden en el refinamiento de la red (Osterman and Overbeek, 2003), (Kharchenko *et al.*, 2004), (Kharchenko *et al.*, 2006), (Chen and Vitkup, 2006), (Green and Karp, 2004), (Kumar *et al.* (2007). Sin embargo, no existen algoritmos que definan criterios de completitud y unicidad para dichos modelos. Esta razón hace aún más engorroso el proceso de refinamiento si se tiene en cuenta que conlleva realizar un estudio de la reversibilidad de las reacciones, análisis de reacciones repetidas dentro del modelo, metabolitos desconectados y reacciones bloqueadas asociadas a los mismos, así como la inclusión de reacciones que fueron estudiadas para otros organismos y que se puede inferir su presencia también para los organismos de estudio.

Teniendo en cuenta esto, y a partir del estudio del primer modelo metabólico de un microorganismo fotosintético, la Synechocystis sp. PCC6803 (Montagud et al., 2010) y de un segundo estudio el de la cianobacteria fotosintética Synechococcus elongatus PCC7942 (Triana et al., 2014) obtenidas por el grupo de Modelización Interdisciplinar InterTech, de la Universidad Politécnica de Valencia, se desarrolló un algoritmo para la herramienta web COPABI que tuvo en cuenta la metodología implementada para la reconstrucción manual de los modelos a escala genómica antes mencionados y que, además, incluye decisiones basadas en criterios probabilísticos proporcionados por los autores, cumpliendo los estándares de coherencia y calidad de los modelos (Reyes et al., 2012).

2.4.1 Metodología que utiliza COPABI

Para la automatización del proceso de reconstrucción de modelos metabólicos a escala genómica en COPABI, se tuvo en cuenta la metodología implementada para la reconstrucción manual de los modelos de la *Synechocystis* sp. PCC6803 (Montagud *et al.*, 2010) y el de la cianobacteria fotosintética *Synechococcus elongatus* PCC7942 (Triana *et al.*, 2014), donde además se incorporan los criterios probabilísticos tenidos en cuenta por los autores.

La siguiente figura refleja los pasos de la metodología implementada a través de los cuales se obtiene un modelo metabólico a escala genómica depurado según los criterios probabilísticos considerados (Reyes, 2013).

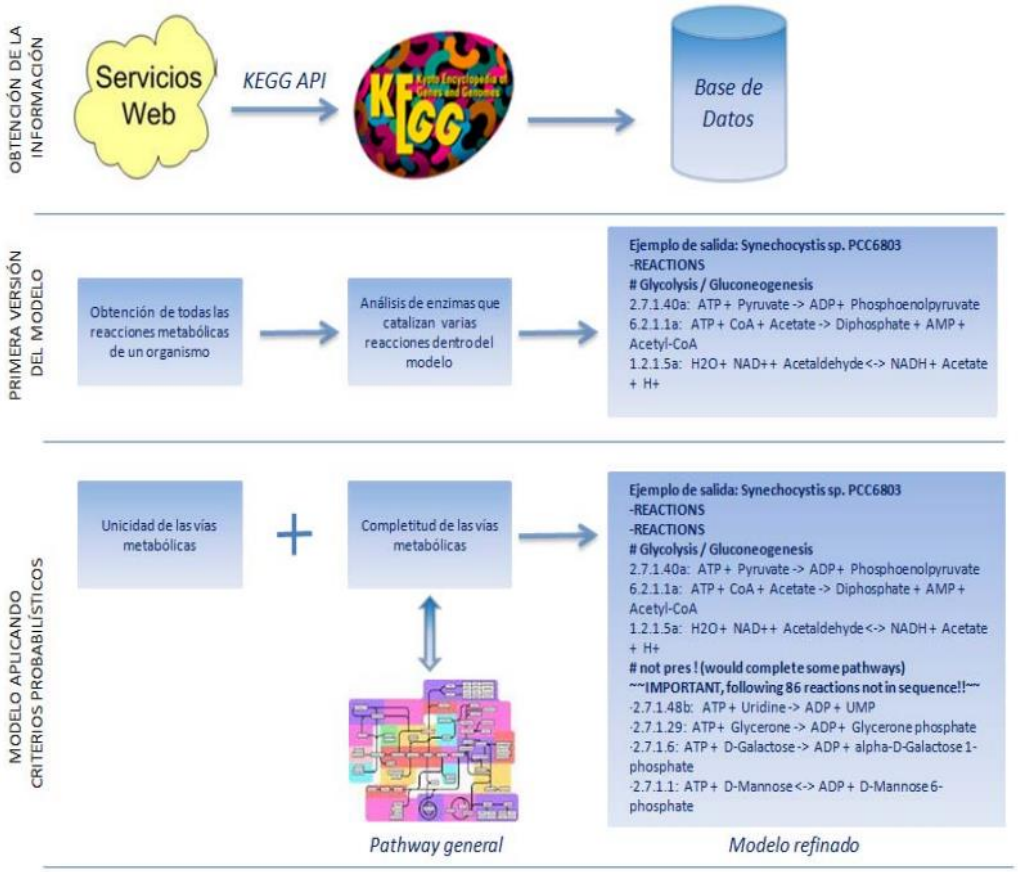


Figura 2.1: Diagrama general de la metodología implementada. Tomado de (Reyes, 2013).

A continuación, se explicarán los pasos en la automatización de dicha metodología (Reyes, 2013):

1. El primer paso consistió en la compilación de todas las reacciones químicas de una ruta en particular o red metabólica en general de un organismo dado. Así mismo, se señalaron el nombre de la vía metabólica y del conjunto de reacciones metabólicas asociadas a él. Además, se tuvieron en cuenta varios elementos a la hora de organizar la información de las reacciones metabólicas.
2. En el segundo paso es donde se identifica el conjunto de enzimas, en el cual cada una de ellas puede catalizar varias reacciones del mismo tipo pues sus sustratos sólo presentan pequeñas diferencias en sus estructuras químicas; se dispone el identificador de la enzima y una letra para denotar que esa enzima cataliza varias reacciones o un número cuando se trata de una reacción espontánea (por ejemplo: 1.1.1.1a, 1.1.1.1b, non-enzymatic1, non-enzymatic2, etc.).

Hasta aquí, se ha obtenido exactamente la información biológica que está almacenada en la base de datos. Posteriormente, se introducen, de manera automática, criterios probabilísticos para satisfacer los criterios de unicidad y completitud exigibles en un mapa metabólico (Reyes, 2013).

2.5 Análisis e implementación de COPABI

Se implementó COPABI, una aplicación web sencilla y de fácil manejo que permite reconstruir modelos metabólicos a escala genómica. COPABI permite exportar la información haciendo uso del SBML (del inglés *systems biology markup language*) (nivel 2, versión 1) (Hucka et al., 2003). En el caso específico de los modelos metabólicos, además del SBML, también se exportan siguiendo los requerimientos de entrada del software OptGene (Patil *et al*., 2005). Ambas formas se utilizan en la identificación de determinados objetivos dentro de la Ingeniería Metabólica. La interfaz de COPABI, de igual manera, permite al usuario introducir un parámetro de decisión dado en forma de un porcentaje, que permitirá valorar la presencia de los metabolitos especificados dentro de la vía metabólica general. Finalmente, el algoritmo implementado permite adicionar al modelo que se construye todas las reacciones dentro de la vía metabólica general que cumplen con las condiciones anteriores de manera simultánea y con el parámetro de decisión definido por el usuario.

La interfaz de COPABI permite al usuario seleccionar, dentro de un listado de metabolitos, aquellos que formarán parte de la Biomasa, además de poder añadir otros.

COPABI presenta la opción de poder conformar la ecuación de biomasa, proporcionando un listado de los metabolitos que pudieran formar parte de la misma, ya sea actuando como reactante o como producto, y con el que el usuario conforma la ecuación de biomasa correspondiente. Además, da la posibilidad al usuario de incorporar nuevos metabolitos que no se encuentran inicialmente en la aplicación (Reyes, 2013).

2.5.1 Mejoras en COPABI

Los modelos metabólicos a gran escala generados por COPABI aún presentan restricciones en torno a deficiencias presentadas en la información biológica, así como por la presencia de reacciones bloqueadas, ciclos fútiles y vías extremas tipo III. Esto influye de forma negativa en los análisis subsecuentes que se realizan utilizando estos modelos. Actualmente existen muy pocos algoritmos que intentan gestionar estas deficiencias, que además lo hacen de forma no intuitiva y centrándose sólo en uno de estos problemas. Por esta razón, se impone aún el uso del trabajo manual que requiere el esfuerzo en tiempo de los especialistas asociados al tema. Teniendo en cuenta tales limitaciones, el trabajo se enfoca en desarrollar un algoritmo computacional que permitiera establecer una metodología de trabajo para gestionar las deficiencias presentes en los modelos metabólicos generados de manera automática.

Chapter 3. PyNetMet: Python tools for efficient work with networks and metabolic models.

3.1 Introduction

Nowadays, the genome-scale reconstruction of metabolic models has become one of the corner stones of systems biology. Reconstructed metabolic models have been used in a wide range of applications, such as the study of metabolism regulation (Barrett *et al*., 2006), (Patil *et al*., 2004), determination of the optimal conditions for growth or prediction of maximum yield of biomass in a determined organism (Edwards *et al* 2001), the search for potential sites for metabolic engineering (Oberhardt *et al*., 2009), the production of biofuels (Montagud *et al*., 2010) and even in the reconstruction of phylogenetic trees (Gamermann *et al*., 2014). One of the most important computational tools for the analysis of metabolic models is the flux balance analysis (FBA) (Varma *et al*., 1993), which consists in the determination of a possible consistent solution for the fluxes in each one of the reactions present in a given model that optimizes some given objective. A particular way to study genome-scale metabolic models is to analyze their underling networks. The simplest example of such networks is to define each metabolite present in a metabolism as a network node, and assign connections in between the nodes based on the connection of the respective metabolites through chemical reactions. Such networks have been thoroughly studied in the literature (Jeong *et al*., 2000), (Rives *et al*., 2003), (Ravasz *et al*., 2002).

Typical genome-scale metabolic models comprise around thousand different metabolites and chemical reactions and, correspondingly, the underlying metabolic networks are complex structures with around one thousand interconnected nodes. Other typical networks studied in systems biology, like protein protein interaction, can be even bigger. The analysis of these complex structures would be nearly unfeasible without the aid of modern computers. There are different available software for performing FBA on metabolic models like the COBRA toolbox, originally developed for MatLab (Schellenberger *et al*., 2011), but now also available for Python, or the OptFlux software (Rocha *et al*., 2010) and also several software for the analysis of networks. Unfortunately, many of the available software have drawbacks.

For instance, there are two different standards for the storage of metabolic models: the SBML (Rocha *et al*., 2010) and OptGene (also known as BioOpt) (Cvijovic *et al.*, 2010) formats, and the available software either use one or another, but not both. On the other hand, some softwares are not free (like MatLab) or are desktop software (like Cytoscape) which limits their uses and integrability with other bioinformatic tools. Also, in order to study different aspects of a given metabolic model, one has to use different software.

Software that perform FBA, which is the case of the COBRA toolbox or the PyCes (Olivier *et al*., 2003) package, do not have the tools to analyze the underlying metabolic network represented by nodes and edges, while software like gephi (Bastian *et al*., 2013), cytoscape (Bauer-Mehren *et*., 2013) or pajek (Jünger et., 2004) that deal with complex networks, do not have the tools necessary to perform flux analysis over metabolic models. Integration between these softwares is extremely difficult when not unfeasible; moreover many of these softwares use different file standards in order to store the models.

In this article we present a series of tools, which have been developed in Python, for dealing with chemical reactions and analyzing networks and metabolic models. Python is a free, open-source, modular, object oriented programming language. Open-source libraries boost the development of bioinformatics by allowing researchers to develop new tools and applications over modules already existent. Moreover, modular programming languages like Python allow easy and efficient integration of its modules with other libraries and software (which is hardly done with desktop applications). In the last years, hundreds of bioinformatic related libraries have been written for Python, like the Biopython package (Cock *et al*., 2009) which contains various standards used in bioinformatics and allows the direct connection with different biological databases, the pysb (Lopez *et al*., 2013), mstacommander (Faircloth *et al.*, 2008) and many others.

The package presented here is called PyNetMet (from Python Network Metabolism), it comprises four classes called Enzyme, Network, Metabolism and FBA. PyNetMet can be downloaded from the Python Package Index (pypi.python.org/pypi/PyNetMet) where one can find installation files for two different operational systems: a windows installation file or a Linux source file which can be used in any UNIX based system (Linux or Mac). In the next section we describe the four classes contained in the PyNetMet package and in section 3 we comment on results obtained by our software in the analysis of three different published genome-scale metabolic models.

3.2 Software Description

The package PyNetMet consists of four classes: Enzyme, Network, Metabolism and FBA, all fully programmed in Python 2.7 language. The class Enzyme has no dependencies; it defines a new type of object that represents a chemical reaction. Class Network has a single dependency (for two specific functions) which is the Python Imaging Package (PIL), for making plots representing the clustering of nodes. The Class Metabolism depends on the classes Enzyme and Network, and class FBA depends on the class Metabolism and on the Python library Pyglpk (which contains tools for solving the associated optimization linear problem).

For a complete list of all attributes and methods of the classes, more detailed examples of use and a short tutorial please refer to the manual that accompanies the PyNetMet

distribution that can be downloaded from the python package index (https://pypi.python.org/pypi/PyNetMet) or the documentation page (pythonhosted.org/PyNetMet/).
In order to use each class, one just needs to import it as a Python module. The examples commented in section 3 can be reproduced following the commands given in the PyNetMet documentation webpage (pythonhosted.org/PyNetMet/example.html) or running the script from the supplementary materials. In our case they were executed in a computer running under a Linux operating system with a icore7 Intel processor and 6 Gb of RAM memory, but the python module is also compatible with windows and mac systems, given that a python interpreter is installed with the necessary above mentioned dependencies.
Next we briefly describe each class in the package and a few definitions and algorithms related to them.

3.2.1 Enzyme

The class Enzyme defines a new type of object that represents a chemical reaction[1]. Enzymes will be the main objects used to build the Metabolism object later on. Its obligatory input is a string containing a reaction written in OptGene format (ex: the string "reac1 : A + 2B -> C" defines an irreversible reaction called reac1 where one molecule of metabolite A interacts with two molecules of metabolite B creating one molecule of metabolite C).
When defining the object one can also give an optional input, also a string, which will be used to indicate the pathway name of a particular reaction. An Enzyme object has attributes that allow an easy verification of the reaction's substrates, products, reversibility, etc.
The class Enzyme has no counterpart in other programming languages or software. It allows the storage of chemical reactions as computer variables such that one is able to sum, subtract or multiply (by numbers) these objects in order to create new (lumped) reactions, or to run buckles over a list of these objects in order to filter them or perform complex analyses.

3.2.2 Network

The Network class defines a graph (collection of nodes and edges) and contains many classical graph theoretical algorithms for its analysis. It should be initiated with one obligatory input and an optional one. The input for this class is the $N \times N$ adjacency matrix which defines the network (N is the number of nodes in the network) and the optional one a list with the node's names. The adjacency matrix, M, is a list of N

[1] Although enzymes are not chemical reactions, enzymes catalyze the chemical reactions and it is a common practice in metabolic modeling to associate each chemical reaction to an enzyme and to name each chemical reaction after this enzyme using, for example, an EC number.

elements, where each element is again a list with *N* elements, each element of this latter list being 0 or 1. If $M[i][j]$ is 1, it means that node i has a directed connection to node *j*. If the matrix *M* is symmetric, the network is undirected, meaning that there is no distinction between a link from node i to j or from node j to i. Otherwise, the network is interpreted as a directed graph, where the connections have an incoming and outgoing node.

Every node in a network can be characterized by some parameters. First, the node's degree is the number of connections it has to other nodes. Another attribute of a node is its clustering coefficient. It is defined by:

$$C_i = \frac{2E_i}{K_i(K_i - 1)} \qquad (1)$$

Where K_i is the degree of node i, and E_i is the number of connections between the neighbors of node i. The average clustering of a network can be calculated straightforward by averaging the values in the list containing the nodes' clustering.

Next, we define the topological overlap (O_{ij}) between two nodes according to (Ravasz *et al*., 2002):

$$O_{ij} = \frac{V_{ij} + \begin{cases} 1 & ,if\ i\ connected\ to\ j \\ 0 & ,otherwise \end{cases}}{\min(n_i, n_j)} \qquad (2)$$

where V_{ij} is the number of common neighbors between nodes i and j and $\min(n_i, n_j)$ is the minimum between the number of neighbors of nodes i and j.

The network's average clustering is an important parameter in characterizing a Network. Different classes of networks have nodes with different tendencies to cluster together defining functional subnetworks (Ravasz *et al*., 2002). Using the nodes clustering coefficients and their topological overlap, one is able to define algorithms that will organize the network's nodes according to their correlation.

In (Ravasz *et al*., 2002) a method for grouping the nodes in clusters is proposed basically by constructing a dendrogram (maximum spanning tree) with the values of the topological overlap matrix (O_{ij}). This tree can be constructed with the Kruskal algorithm, which is implemented in the Network class. Another interesting method for ordering the nodes is proposed in (Rybarczyk-Filho *et al.,* 2011). Although this later method has many improvements with respect to the dendrogram one, it is based on a Monte-Carlo simulation and given the size of the networks and the amount of Monte-Carlo steps needed in order to perform a proper simulation, the computation is usually costly. Here we propose a yet different method which is computationally more efficient and returns results at least as good as the dendrogram method.

The objective of the following algorithm is to reorder the nodes in the adjacency matrix (or the topological overlap one), such that nodes close to each other are correlated in the sense that they share neighbors which are also correlated among them, obtaining in this way an ordering where nodes belonging to common clusters are nearby to each other. So the output of this algorithm will be a list with a new ordering of the nodes. The algorithm follows the following steps:

Choose any node i to start with. Add it to the ordering.

From node i, find the node j for which x_{ij}^2 defined below is minimum:

$$x_{ij}^2 = \sum_{K \in E'} \frac{1}{\max(\in, C_k)} \left(\frac{O_{ik} - O_{jk}}{O_{ik} + O_{jk}} \right)^2 \qquad (3)$$

where E' is the set of all nodes that have not been added to the ordering.

Add node j to the ordering.

Set node j as i and repeat the process from step (2) until the set E' is empty.

The use of ϵ and the function $\max(\in, C_k)$ is to avoid a division by zero in the case that node k has clustering coefficient equal to zero and so to have a stable algorithm. The parameter ϵ should be very small (smaller than the value of the smallest non-zero C_k in the network). In performing the simulations the value is set to $\epsilon = 0.00001$, changing this value, as long as it is sufficiently small, has no effect in the obtained ordering. The algorithm is implemented in the Network's class method plot_nCCs. Figure 3.1 shows examples of plots obtained for three metabolic models by three different ordering algorithms: without any ordering, with the Kruskal algorithm and with the algorithm described here.

3.2.3 Metabolism

This class defines an object with a full metabolic model. The metabolic model can be given as input in three different ways. By default one can use a single input which is a string containing the file name (with path) of a metabolic model in OptGene format. Alternatively, one can use a file in SBML format and finally one can define lists containing reactions, constraints, external metabolites and objective function directly from the command line and use them as input for the class. So, this class works either as a parser for OptGene or SBML file formats or as a platform to construct new metabolic models from scrap.

This class has also the dump method that allows one to write an output file with the stored model either in OptGene or SBML file formats. This resource allows the class to

be used as a translator between OptGene and SBML file formats, for one can load the model in one format and dump it in the other format. This feature is an advantage with respect to most software used in metabolic analysis, which are usually compatible with only one format for the metabolic models.

When parsing a SBML file, the PyNetMet package will look for the SBML tags <reaction> and <species> and inside each element of <reaction> it will generate a list of substrates and products based on the elements defined by the tags <listOfReactants> and <listOfProducts>. As long as different versions of SBML maintain these tags as standards, and their attributes, the current version of PyNetMet will be able to read SBML. If these standards are changed in the future, a new version of PyNetMet will be needed in order to keep its functionality as a SBML parser.

The main attribute from this class is its enzymes list, which contains all chemical reactions in the model. This list can be altered either directly (which is not advisable since other attributes of the class will not be automatically updated unless one calls the calcs method afterward), or by making use of the bad_reacs, add_reacs and pop methods. The possibility to change the enzymes list, gives a new functionality to the class Metabolism, namely it can be used as a platform to produce in silico mutants and perform metabolic engineering studies and simulations. The use of this class together with the Network and FBA classes offers rich resources for an extensive analysis of any metabolic model.

3.2.4 FBA

The FBA class offers tools for performing flux simulations and analysis of a metabolic model. It has methods defined which are based on the FBA for studying essential reactions, sensibility of the objective function with respect to any given reaction, comparison of different realizations of the FBA, among others.

To call this class one must give one obligatory input, which is a Metabolism object with a metabolic model. It can also receive two optional inputs which are the precision (eps, value under which a flux is considered zero which by default is set to 10^{-10}) and a choice of maximizing or minimizing the objective the default choice is maximize).

3.3 Applications

In this section we exemplify some uses of our tools by analyzing real metabolic models taken from the literature. We chose three models to work with, the first is the *iSyn811* model of *Synechocystis sp*. *PCC6803* (Montagud *et al*., 2010). The second is the metabolic model *iCM925* for the organism *Clostridium beijerinckii NCIMB 8052* (Milne *et al*., 2011) and last is the model *iAK692* for *Spirulina platensis C1* from (Klanchui *et al*., 2012). All these models are available from the journals as supplementary materials, the first one in OptGene format and the other two in SBML format. These have been

downloaded and saved in a working folder and can be directly accessed by PyNetMet tools.

With a few lines of code, one is able to do complex analysis of any metabolic model (see supplementary materials). We present in table 1 the results of calculating the average clustering of the networks representing each one of the metabolic models, the number of disconnected metabolites in each model and the average distance between any two metabolites in the network.

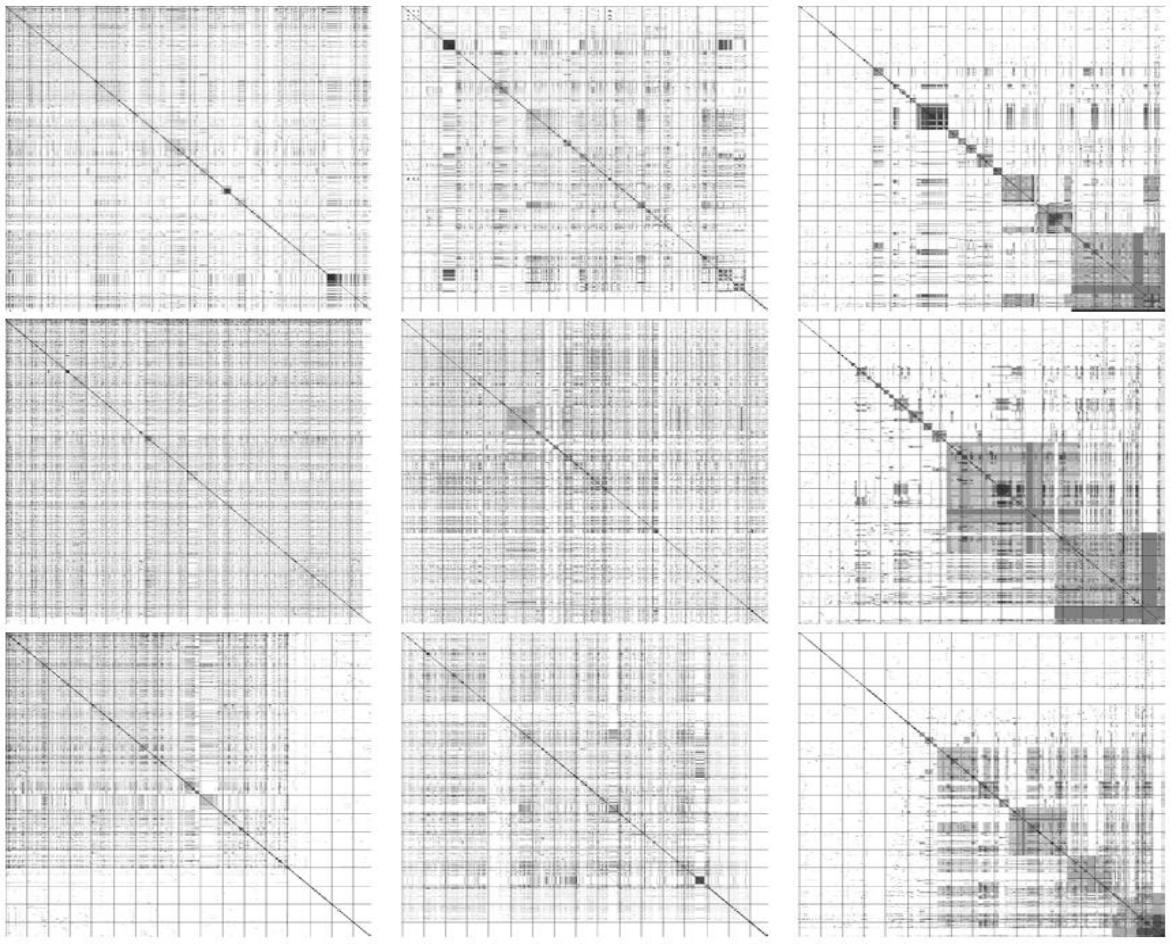


Figura 3: Plots for the topological overlap of metabolites. The first, second and third rows refer to the plots obtained from the three models analyzed: iSyn811, iCM925 and iAK692, respectively. The plots in the first column are for an arbitrary ordering of the metabolites, in the second column an ordering is obtained via the Kruskal algorithm and in the third column the ordering is obtained by the algorithm implemented in the plot_nCCs method of the Network class.

The two network averages shown in table 1 show two important features of biological networks that differentiate them from random networks and other kinds of networks, namely their relatively high (tough disperse) average clustering and the small world (Amaral *et al*., 2000) property that refers to the fact that the average distance between two nodes scales very slowly with the size of the network.

Model	$\bar{C}$	σ_C	*#D.M.*	$\bar{\ell}$	σ_ℓ	*Essential*
iSyn811	0.166	0.224	13	3.248	1.161	221
Icm925	0.245	0.244	2	2.790	0.703	166
Iak693	0.200	0.244	42	3.053	1.001	249

Tabla 1: Results from the models analysis. The network parameters were calculated for the undirected version of the networks obtained from connecting metabolites that appear as substrate and product in the model's reactions.The $\bar{C}$ column indicates the average clustering coefficient, the #D. M. column indicates the number of nodes disconnected from the main component of the network, the $\bar{\ell}$ column indicates the average distance between two nodes in the network and the column Essential has the number of essential reactions for producing the objective function. The σ's are the standard deviations for the two averages.

To show this, we generated 100 random networks with the same number of nodes and links that the synechocystis metabolic network and we show in table 2 a comparison between the random networks and the iSyn811 network. To create the random networks a set of nodes is generated. Then, two nodes are selected at random and connected. The random connecting process is repeated until the network has the desired number of links. From table 2 it is clear that the average clustering in random networks is basically zero and the average distance between two nodes is two times larger than in metabolic networks.

Model	$\bar{C}$	σ_C	$\bar{\ell}$	σ_ℓ
iSyn811	0.166	0.224	3.248	1.161
Random	0.005	0.043	6.213	1.557

Tabla 2: Comparison between metabolic and randomnetworks. Comparison between the clustering coefficient and average distance between two nodes in the metabolic network of the Synechocystis sp. PCC6803 and the average of 100 random networks.

Apart from the network analysis of the models, one can use the methods in class FBA. Just by calling the class with a metabolic model as input one can directly obtain the FBA result by printing the class object. The result is a string listing the reaction names and their respective fluxes ordered, by default, according to these fluxes.

A straight forward method to analyze a FBA is the essential method, which checks if a reaction is essential for producing flux in the objective function. When called, it returns one Boolean value stating if the reaction is essential or not and a second value which informs of the relative change in the objective flux with the input reaction removed. In table 1 one can also find the number of essential reactions in each metabolic model.

Other methods to analyze a FBA are the shadow and max_min methods which work in a similar way. In each method one has to use as input an integer indicating a reaction number. The shadow method has two other optional inputs, the first one indicates the change in the original flux in order to calculate the derivative (the result should be independent of this choice, since the problem is linear) and the second indicates if one wishes the relative change or the absolute change.

The method max_min has the fixobj optional input. Its algorithm fixes the flux in the objective function to its original value times the value in fixobj (this should always be a value between 0 and 1). Then it sets the input reaction as objective and minimizes it, then maximizes it in its natural direction ($S \longrightarrow P$) and then again minimizes it and maximizes it in the reversed direction ($S \longleftarrow P$). It returns a two element list, each element is a tuple with the value of the flux in the reaction minimized and maximized in the direct direction and in the reversed direction, respectively. If the reaction is irreversible or there was no feasible solution, it returns the string "X". This method can be used in analysis of maximal and minimal throughput of a reaction. So, it determines limits in knockdowns or over-expression, since it shows the flux range at which a determined reaction can occur. Since a reaction rate is proportional to the expression of some enzyme, suppression or over-expression of the correspondent gene will have direct (proportional) effect over the reaction flux. Instead of using this method to determine limits, it is also possible to directly redefine the constraints of a given reaction (altering the values in fba.constr) in order to fix the flux of a given reaction.

The implementation of these examples, which do not intend to exhaust the uses of the PyNetMet tools and their functionality, can be found in the PyNetMet documentation webpage (pythonhosted.org/PyNetMet/example.html).

Capítulo 4. BioMOE, Herramienta de análisis multiobjetivo.

4.1 Introducción

A menudo analizar los modelos metabólicos partiendo de la optimización monoobjetivo no siempre se acerca todo lo deseado a la realidad, puesto que uno o más objetivos pueden entrar en conflicto porque tienen como denominador común la necesidad de elegir entre diferentes alternativas que han de evaluarse en base a diferentes criterios.

Para este fin existen un grupo de métodos que podemos afirmar que son clásicos, que combinándolos con otros métodos se pueden obtener análisis más profundos y simulaciones más exactas.

En este capítulo analizaremos algunos conceptos como los de Algoritmos Evolutivos y Optimización Multiobjetivo. Veremos el Algoritmo Evolutivo de Optimización Multiobjetivo que sigue la aplicación BioMOE para determinar un grupo de soluciones donde ninguna es mejor que otra. Mostraremos también los principios de implementación y las vistas fundamentales de la herramienta.

4.2 Optimización Multiobjetivo

De modo general podemos definir un problema de optimización multiobjetivo de la siguiente manera:

$$mín_{\alpha} J(\alpha) = [J_1(\alpha), \dots\dots\dots, J_n(\alpha)] \qquad (1)$$

Sujeto a las restricciones:

$$g(\alpha) \geq 0 \qquad (2)$$

$$h(\alpha) = 0 \qquad (3)$$

$$\iota_{\alpha_i} \geq \alpha_i \geq \mathcal{u}_{\alpha_i}, \qquad i = [1, \dots\dots\dots, n] \qquad (4)$$

donde α es el vector de decisión, de dimensión $\dim(\alpha) = \mathrm{n}$ y $J(\alpha)$ es el vector de los objetivos de dimensión $\dim\big(J(\alpha)\big) = m$; $g(\alpha)$ y $h(\alpha)$ son vectores de restricción de igualdad y desigualdad respectivamente y ι_{α_i} y $\mathcal{u}_{\alpha_i}$ son los limites superior e inferior del espacio vectorial de la variable α_i (Reynoso-Meza et al., 2014).

Si en lugar de minimizar se desea maximizar algún objetivo no hay más que tener en cuenta que:

$$max_{\alpha}\, J_i(\alpha) = \; -m\acute{\imath}n_{\alpha} - \, J_i(\alpha) \qquad (5)$$

Para buscar diferentes soluciones y posteriormente seleccionar la más convincente, se generan muchas soluciones óptimas en un Frente de Pareto (Mezura-Montes et al., 2008); un vector de objetivos $J(\alpha^1)$ es óptimo en el sentido de Pareto si no existe otro vector de objetivos $J(\alpha^2)$ tal que $J_i(\alpha^2) \leq J_i(\alpha^1)$ para todos $i \in [1, \dots\dots, m]$ y $J_j(\alpha^2) < J_j(\alpha^1)$ para al menos un $j, j \in [1, \dots\dots, m]$.

Todas las soluciones del Frente de Pareto se dice que son soluciones no dominadas; si tenemos muchas soluciones y deseamos seleccionar las mejores soluciones podemos aplicar:

Dominancia estricta (Mezura-Montes et al., 2008). Un vector de objetivos $J(\alpha^1)$ es dominado por otro vector de objetivos $J(\alpha^2)$ si $J_j(\alpha^2) < J_j(\alpha^1)$ para todos los $i \in [1, \dots\dots, m]$.

Dominancia (Mezura-Montes et al., 2008). Un vector de objetivos $J(\alpha^1)$ domina a otro vector de objetivos $J(\alpha^2)$ si $J(\alpha^1)$ no es peor que $J(\alpha^2)$ en todos los objetivos y es mejor en al menos un objetivo. Esto se denota por $J_j(\alpha^1) < J_j(\alpha^2)$.

Dominancia débil (Mezura-Montes et al., 2008)). Un vector de objetivos $J(\alpha^1)$ domina débilmente a otro vector de objetivos $J_j(\alpha^2)$ si, $J_j(\alpha^1)$ no es peor que $J_j(\alpha^2)$ en todos los objetivos.

4.3 Algoritmos evolutivos de optimización multiobjetivo

Una rama de la inteligencia artificial son los algoritmos evolutivos (AE); éstos son métodos de optimización y búsqueda estocásticos que se inspiran en la genética y en los principios establecidos por Darwin en 1859 sobre la evolución de las especies en el mundo biológico.

Estos algoritmos se modelan computacionalmente simulando la selección natural y el entrecruzamiento de las especies por medio de la recombinación genética y la mutación. En ellos se mantiene un conjunto de entidades que representan posibles soluciones, las cuales se mezclan y compiten entre sí, de manera que las más aptas son capaces de prevalecer a lo largo del tiempo, evolucionando hacia mejores soluciones.

Existen varias técnicas evolutivas y bioinspiradas, habitualmente usadas en los algoritmos evolutivos de optimización multiobjetivo. Las más populares incluyen algoritmos genéticos, optimización por enjambre de partículas (Particle Swarm Optimization, PSO) y evolución diferencial, aunque cada vez son más comunes técnicas bioinspiradas como el algoritmo de colonia de abejas o la optimización de colonia de hormigas (Reynoso-Meza et al., 2014).

En (Siurana *et al*., 2017) se presenta un algoritmo evolutivo de optimización multiobjetivo que se aplica para análisis de microrganismos como Synechocystis sp. PCC6803. El algoritmo de optimización multiobjetivo empleado consiste en una adaptación del algoritmo sp-MODE (Reynoso-Meza et al., 2014, Reynoso-Meza et al., 2010), desarrollado en el seno del Grupo de Control Predictivo y Optimización Heurística (CPOH) perteneciente al Departamento de Ingeniería de Sistemas y Automática (DISA) de la Universidad Politécnica de Valencia.

Este algoritmo está implementado en MatLab y se puede utilizar para otros tipos de problemas de optimización multiobjetivo. En este capítulo proponemos una herramienta en Visual Studio .Net (C#) que nos permita hacer análisis y simulaciones en microrganismos con más de un objetivo que nos interese, con algunas modificaciones al algoritmo propuesto por (Siurana *et al*., 2017).

4.4 Descripción del algoritmo

I. Generar población inicial (P_o) de manera aleatoria añadiendo puntos ancla (Siurana *et al*., 2017).

- Calcular los puntos ancla, que se definen como aquellos puntos que se obtienen al optimizar el valor de cada uno de los objetivos independientemente. Es decir, aquellos que son solución del problema calculados como los óptimos para cada objetivo individualmente utilizando el método Simplex.
- Para construir la población inicial debemos crear cada vector (V) que contiene esa población inicial.
- Para crear un vector debemos generar de manera aleatoria uno a uno los flujos (Fl_j) de ese vector. Para ello se crea una variable random (Rd), que sigue una distribución uniforme entre 0 y 1 y, tomando en cuenta los límites inferiores (Lb_j) y superiores (Ub_j) de cada flujo, se calcula como un valor aleatorio, como se indica en la ecuación (3).
- A esa población inicial se le incluyen las anclas que fueron calculadas por el método simplex.

$$P_o = V_1 \dots\dots\dots V_p \qquad (1)$$

$$V_k = Fl_1 \dots\dots\dots Fl_n \qquad (2)$$

$$Fl_j = \left(Ub_j - Lb_j\right) * Rd + Lb_j \qquad (3)$$

$$\text{para todos los } j \in [1, \dots\dots, n] \qquad (4)$$

II. Evaluar los costes de la población inicial (Siurana *et al*., 2017).

- A cada individuo[2] se le asociarán tantos costes como objetivos se hayan definido; por ejemplo, si se está trabajando con tres objetivos, cada individuo tendrá asociados tres costes.

[2] Se le denomina individuo a una distribución de flujo obtenida.

- Cada uno de esos costes depende del valor de flujo de la reacción correspondiente más una penalización, si viola la restricción de estado estacionario.
- El coste (C) de un individuo se toma como el valor de los objetivos en ese vector más una penalización (Pen), por lo que si tiene tres objetivos tendrá tres costes.
- Si un objetivo se minimiza, el coste inicial es el valor de flujo de la reacción correspondiente dentro del vector; si se maximiza, el coste inicial sería el valor del flujo multiplicado por (-1).
- La penalización será igual a cero si el vector cumple que $S * V = 0$ (restricción de estado estacionario). En caso contrario, el vector se penaliza asignando a cada coste el valor correspondiente al punto nadir[3] (Na) más la desviación respecto del estado estacionario (el sumatorio en valor absoluto de las coordenadas de $S * V$).
- Esos valores se almacenan en una matriz de costes (Mc_{pxq}) que tendrá q columnas, igual a la cantidad de objetivos del problema, y p filas, que serán iguales a la cantidad de individuos que tenga la población.

$$C(V_k) = \begin{cases} Fl_j\ (-1)^{\propto} & si\ Pen = 0 \\ C(Na) + Pen_k & si\ Pen\ > 0 \end{cases} \qquad (5)$$

$$donde:\ Pen_k = \sum_{j=1}^{n} abs\ (S * V_k) \qquad (6)$$

$$\propto = \begin{cases} 0\ si\ min \\ 1\ si\ max \end{cases} \qquad (7)$$

$$Mc_{pxq} = \begin{bmatrix} C_{11} & \cdots & C_{1q} \\ \vdots & \ddots & \vdots \\ C_{p1} & \cdots & C_{pq} \end{bmatrix} \qquad (8)$$

III. Aplicar filtro de dominancia a la población inicial (Siurana *et al.*, 2017).

III.1 Comprobar si el individuo está dominado.

[3] Se le denomina nadir al peor valor que hay en el Frente de Pareto para ese objetivo.

- En la matriz de costes se toma cada uno de los individuos y se compara con el resto; si todos los costes de un individuo $C(V_a)$ son mayores que los de otro individuo $C(V_b)$, entonces el vector $C(V_a)$ está dominado por el vector $C(V_b)$.

III.2 Seleccionar todos los que no estén dominados y guardarlos en otra matriz donde estarán los individuos élite.

- Para cada individuo hay que guardar también sus costes, por lo que tendremos una matriz con los vectores completos (individuos élite) y otra con los costes de esos individuos.

IV. Guardar en el vector de parámetros los extremos del frente (la élite) en ese momento (nadir (Na) y utopía[4] (Up)) (Siurana *et al*., 2017).

- Para obtener el nadir, se toma el peor valor (el de coste máximo) para cada objetivo entre los valores de ese objetivo en el frente de Pareto.
- Para calcular utopía, se toma el mejor valor (el de coste mínimo) para cada objetivo entre los valores de ese objetivo en el frente de Pareto.

Bucle (la condición de parada está en el número de iteraciones -generaciones- que desee el usuario):

V. Generar la población (Siurana *et al*., 2017).

- La mitad de la población (si existen suficientes individuos) estará compuesta por individuos de la élite (escogidos aleatoriamente) más las anclas.
- El resto se completa de la población anterior tomando los vectores de manera aleatoria.

VI. Generar los mutantes (Siurana *et al*., 2017).

- Los mutantes los generamos tomando tres vectores de manera aleatoria V_{r1}, V_{r2} y V_{r3}, donde a V_{r1} le sumamos un factor de escala (F) por la diferencia entre V_{r2} y V_{r3}.

[4] Se denomina utopía al mejor valor (el de coste mínimo) para cada objetivo entre los valores de ese objetivo en el frente de Pareto.

- Comprobamos que los mutantes estén dentro de los límites y, si no lo están, se ajustan los flujos al extremo más cercano.

$$V_k = V_{r1} + F * (V_{r2} - Vr_3) \qquad (8)$$

$$k \neq r1 \neq r2 \neq r3 \qquad (9)$$

VII. Tomar como hijos los mutantes generados (dado que el crossover rate es 1) (Siurana *et al*., 2017).

VIII. Evaluar los costes y penalizaciones de los hijos (Siurana *et al*., 2017).

IX. Crear la nueva población (Siurana *et al*., 2017).

- Para crear la nueva población se usa la dominancia estricta entre cada padre y su hijo: sólo se tomará el hijo en el caso de que sea mejor para todos los flujos.
- Siempre se mantiene el tamaño final de la población.

X. Aplicar el filtro de dominancia a los hijos junto con la élite (Siurana *et al*., 2017).

- Se toman los hijos con toda la élite y se les aplica el filtro de dominancia para generar una nueva élite (siempre tenemos que tener la élite guardada y se actualiza con cada iteración).

Fin del Bucle

XI. Devolver los vectores que conformán la élite

4.5 Descripción de la aplicación

La herramienta BioMOE es una herramienta hecha en Visual Studio que consta de seis clases: “Read”, “Species”, “Reactions”, “Stoichiometry”, “FBA” y “AE”, y de una interfaz visual que le permite al usuario interactuar con la herramienta sin la necesidad de utilizar comandos. La clase “Read” depende de las clases “Species” y “Reactions”. Ésta es la encargada de realizar la entrada de la información de los modelos que pueden tener origen de datos en un fichero OptGene o uno sbml; a medida que toma los datos los va guardando mediante listas de objetos de las clases “Species” y “Reactions”. Las clases “Species” y “Reactions” no tienen dependencia de ninguna clase. Una vez que

tengamos las listas de las especies y reacciones, la clase “Stoichiometry” construye una estructura para guardar la matriz estequiométrica del organismo. La clase “FBA” depende de las clases “Species”, “Reactions” y “Stoichiometry” que contienen los datos necesarios para que con la librería GlpkSharp (que contiene herramientas para resolver la optimización lineal asociada al problema) determinar las anclas.

Para que la clase “AE” pueda implementar el algoritmo evolutivo, la clase “FBA” tiene que ejecutarse una cantidad de veces igual a la misma cantidad de objetivos que se deseen optimizar, por lo que es dependiente de la clase “FBA”.

4.5.1 Clase “Read”

En esta clase se crean los campos para guardar la información general del modelo metabólico como el id y el nombre del modelo; además, en caso de ser un fichero sbml, se guarda el nivel y la versión de sbml.

En esta clase se lee todo el fichero de origen de datos y se crean listas de tipo “Species” para guardar los metabolitos internos y los externos que tiene el modelo. También se crea una lista de tipo “Reactions” en la cual se guardarán todas las reacciones del organismo.

4.5.2 Clase “Species”

En la clase “Species” se crean los metabolitos. Presenta los campos de id de un metabolito, nombre de ese metabolito y el coeficiente estequiométrico que puede tener ese metabolito en una reacción determinada. Cuando se construye un objeto de esta clase, se le asigna por defecto siempre el valor de 1, quepuede cambiarse por las propiedades de ese campo en la clase.

4.5.3 Clase “Reactions”

En la clase “Reactions” se construye una reacción con todas las características de las reacciones. Tiene campos para el id y el nombre de la reacción y para guardar el límite superior e inferior de la reacción por los que se conoce si es reversible o irreversible. También tenemos una lista de reactantes y otra de productos; en cada una se guardan los metabolitos que conforman los reactantes y los productos de la reacción.

En esta clase existen funciones que se llaman desde la clase “Read” que le pasan una cadena de texto y la desglosan para determinar los límites de la reacción, la reversibilidad, lista de reaccionantes y de productos y la estequiometría de cada metabolito para esa reacción.

4.5.4 Clase “Stoichiometry”

“Stoichiometry” es una clase para crear una estructura que necesita la librería GlpkSharp. Esta clase tiene un campo entero fila, uno entero columna y un campo **doublé** valor. Esta clase está creada para gestionar la información de la matriz estequiométrica; como es una matriz poco densa, en la que sus valores mayormente son cero, sólo se guarda en un arreglo el número de la fila y la columna donde hay un valor distinto de cero y el valor que tiene esa posición. Esta clase depende de las clases “Species” y “Reactions”, ya que cuando se construyen las listas de reacciones y de metabolitos entonces se pasa a llenar el arreglo de tipo “Stoichiometry”. Una vez construidas las listas de reacciones y metabolitos, y con la información de la estequiometría, la aplicación nos muestra la información del modelo como se puede ver en la figura 4.1.

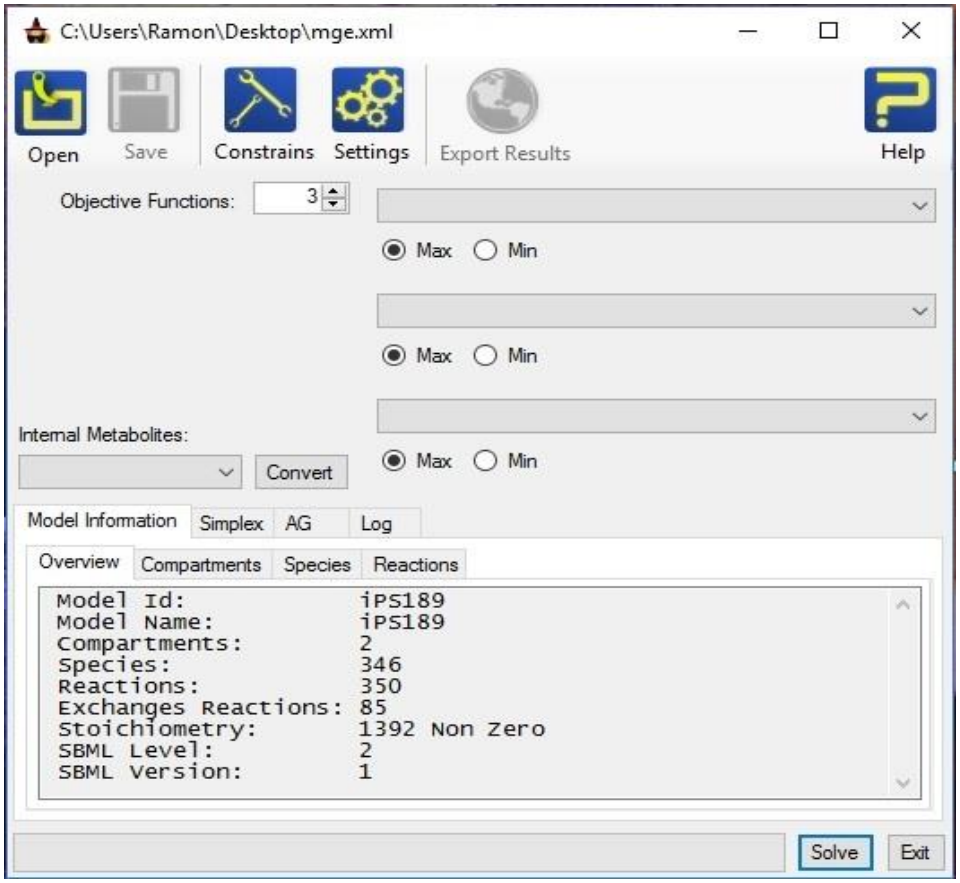


Figura 4: Información del modelo iPS189.

4.5.5 Clase "FBA"

Las anteriores clases son preparatorias para poder comenzar a ejecutar el Algoritmo Evolutivo de Optimización Multiobjetivo. En esta clase tenemos las listas de reacciones y metabolitos, el arreglo de "Stoichiometry" y los objetivos que deseamos optimizar.

En esta clase se toma el arreglo de "Stoichiometry" y se guarda en tres arreglos diferentes, donde se hace una llamada a los metodos "LoadMatrix" y "SetRowBounds" de la GlpkSharp para cargar la matriz estequiométrica y los límites de cada reacción. Con el método "ObjectiveDirection" se maximiza o minimiza, según se desee.

Como se muestra en la figura 4.2, podemos optimizar linealmente cada uno de los objetivos por separado y tendremos un vector por cada objetivo con el valor óptimo de ese objetivo. Estos vectores los consideramos como las anclas del algoritmo y pasarán a formar parte de la población inicial y del grupo élite de soluciones del Frente de Pareto.

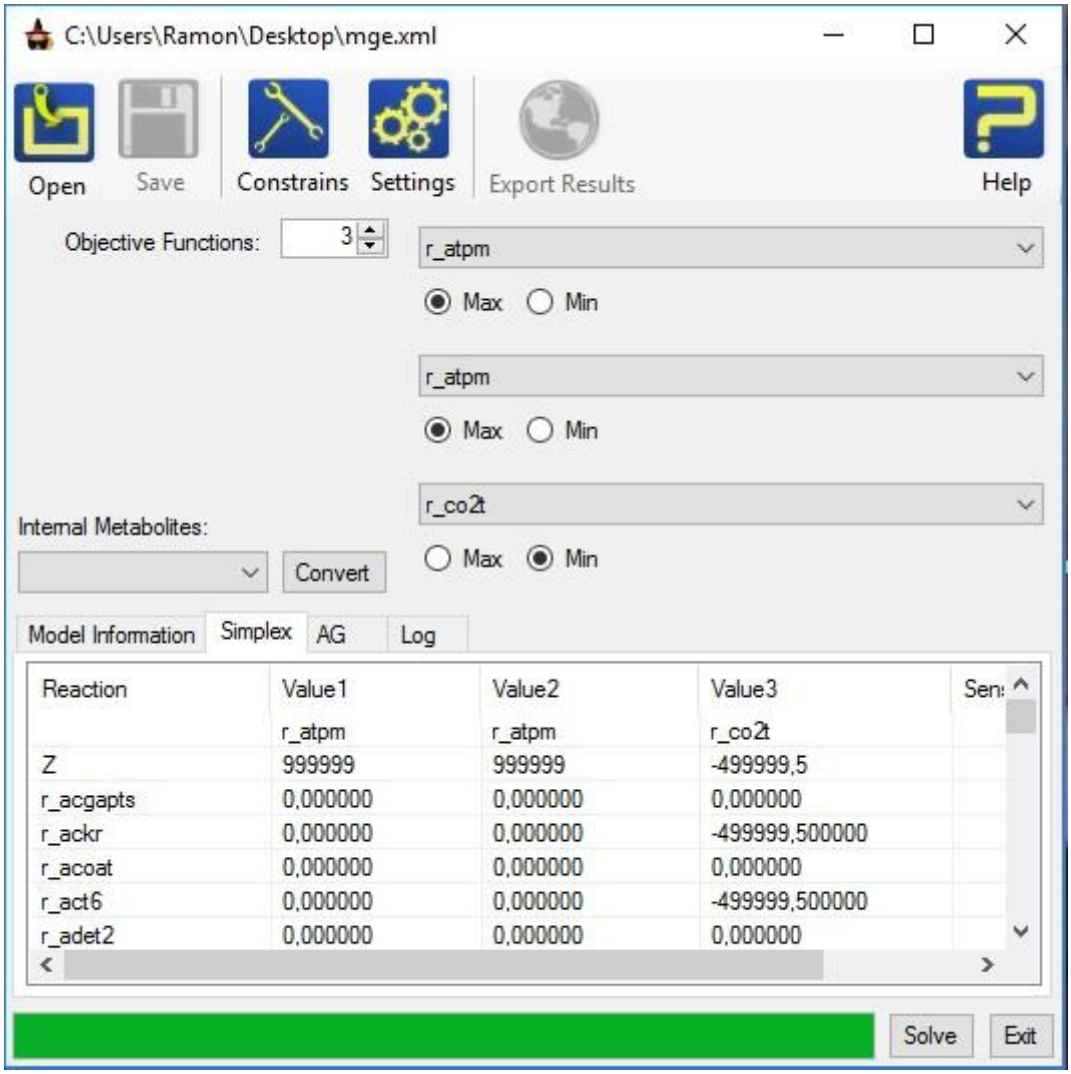


Figura 5: Anclas seleccionadas que se calculan con la librería GLPKsharp.

Antes de comenzar las iteraciones en el algoritmo evolutivo, la aplicación le permite al usuario poder ajustar los datos mediante la interfaz de la figura 4.3.

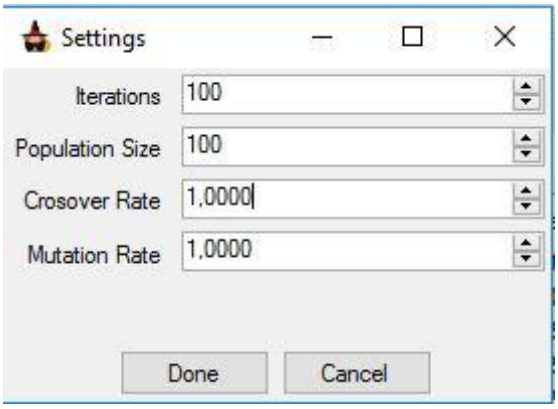


Figura 6: Ajustes del Algoritmo Evalutivo.

4.5.6 Clase “AE”

Teniendo las anclas del algoritmo calculadas, se pasa a generar la población inicial. Con una variable “random” en el método “Initialisation”, generamos de manera aleatoria una población inicial de 100 vectores incluyendo las anclas. Se verifica que cada valor de este vector esté dentro de los límites de la reacción y, en caso contrario, se lleva el valor a esos parámetros.

Una vez generada la población inicial, se evaluarán los costes en el método “Evaluation”. Para cada flujo se comprueba si cumple con la restricción de estado estacionario y, si no, se penalizan los costes para llenar la matriz de costes con los costes de cada vector en la población.

El filtro de dominancia se aplica a toda la población para determinar qué vectores son los dominados y poder situar en la élite a los vectores que no son dominados. A estos individuos que pertenecen a la élite se les guarda también sus costes. Este filtro de dominancia se realiza en el método “Dominance_filter” que retorna los individuos de la élite y los costes de la élite.

Así mismo, los valores de nadir y utopía se calculan para cada iteración.

Con toda esta información, se comienza a iterar la cantidad de veces que el usuario declare como condición de parada y, por cada iteración, se va a generar una población de la cual se tomaran los mejores vectores para que compitan en la élite.

Para generar la población en el método “Generate” se toma la mitad de los individuos de la élite más las anclas y la otra mitad se toma de la población anterior, toda la selección se hace de manera aleatoria.

Con esta población se procede a generar los mutantes con el método “Crossover”, los cuales se toman como hijos ya que el grado de mutación que se busca es alto; además, se vuelven a evaluar los costes y penalizaciones.

Para generar la población con el método “Selection_children”, se determina cuál de los vectores seleccionamos para la nueva población. La manera de hacerlo es seleccionar el

vector que domine y, en caso de que ninguno domine al otro, se selecciona para la nueva población al padre.

Finalmente, se aplica nuevamente al filtro de dominancia a la población para actualizar la élite de los individuos. En la figura 4.3 se muestran los datos de la iteración número 20.

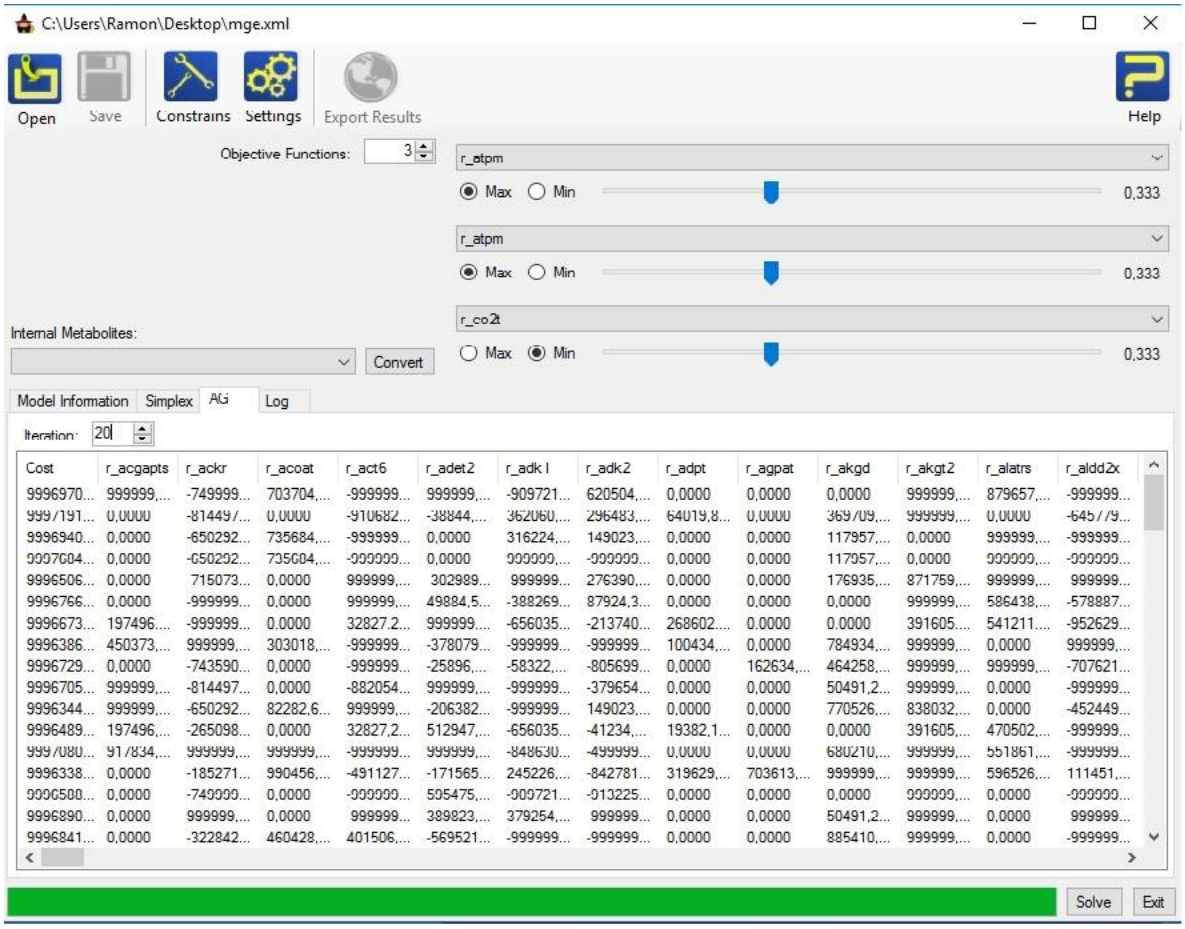


Figura 7: Datos de la iteración número 20.

Se generarán nuevas poblaciones hasta que se llegue al final de la cantidad de iteraciones señaladas por el usuario.

Capítulo 5. CompNet, Herramienta para la comparación de dos modelos metabólicos.

5.1 Introducción

Existen disímiles herramientas para el análisis de modelos metabólicos; sería de interés poder comparar la estructura de diferentes organismos, por ejemplo, para seleccionar las especies o las cepas que mejor se adapten a una aplicación en concreto. En el caso del uso de cianobacterias como plataformas de producción, resulta de gran utilidad caracterizar el metabolismo asociado a la producción de hidrógeno: saber si son o no capaces de producirlo, clasificarlas según el tipo de hidrogenasa que emplean, determinar si el hidrógeno se produce como consecuencia de algún otro proceso metabólico como algunos casos en que está asociado a la fijación de nitrógeno, etc. Comparar la estructura metabólica de distintos organismos para estudiarlos a nivel evolutivo, analizar qué rutas metabólicas derivan de otras predecesoras, en qué punto se bifurcan dos ramas evolutivas en reacciones distintas que cumplen la misma función, e incluso se pueden construir árboles filogenéticos en base a similitudes y diferencias metabólicas (Navarro *et al*. 2009) (Tamagnini *et al*. 2007) (Tiwari *et al*. 2012).

Para poder comparar un modelo metabólico con otro es necesario hacer una analogía entre modelos metabólicos y grafos. Analizando la composición de un modelo metabólico es necesario introducir otros conceptos como el de una Red de Petri que fue introducida por Carl Adam Petri (Petri *et al.* 1966) para la descripción formal de sistemas cuya dinámica se caracteriza por concurrencia, sincronización, exclusión mutua y conflictos.

En este capítulo presentamos una herramienta que, basándose en estos conceptos, permite comparar dos modelos metabólicos con el objetivo de conocer las semejanzas y diferencias que hay entre ambos. A partir del uso de tres métricas, podemos saber qué tan iguales o diferentes son y cuáles son los cambios necesarios en un modelo para que sea similar a otro.

5.2 Igualdad entre dos modelos metabólicos

Es importante aclarar que en nuestro caso no debemos utilizar isomorfismos en el momento de comparar dos modelos metabólicos, pues lo que se busca es una relación de igualdad y el isomorfismo lo que nos ofrece es una relación de equivalencia.

Para que dos grafos sean iguales han de tener, primero, exactamente los mismos vértices y las mismas aristas. Observemos los dos grafos de la figura 5.2, que cumplirían las condiciones de equivalencia para que fueran isomorfos.

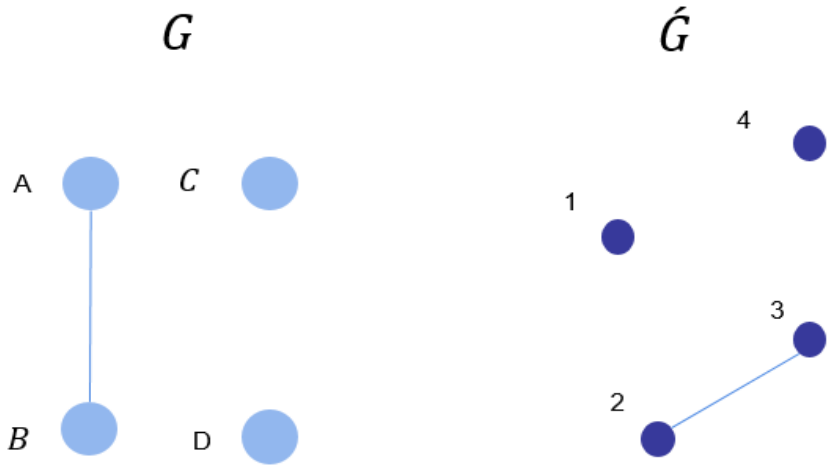


Figura 8: Grafos que cumplen con un isomorfismo.

Tienen vértices distintos: los del de la izquierda son *{A, B, C, D}* y los del de la derecha, *{1, 2, 3, 4}*, y cada uno tiene una única arista. Son grafos distintos, simplemente porque tienen vértices distintos.

Para comparar dos modelos metabólicos, se representan como Redes de Petri. En todas las reacciones (transiciones) coincidentes, se puede asegurar que los metabolitos (vértices) que las componen coinciden plenamente. Y en aquellas en las que las reacciones no coincidan, tendríamos que buscar cuáles se parecen más, atendiendo a los metabolitos contenidos, como se muestra en la figura 5.3.

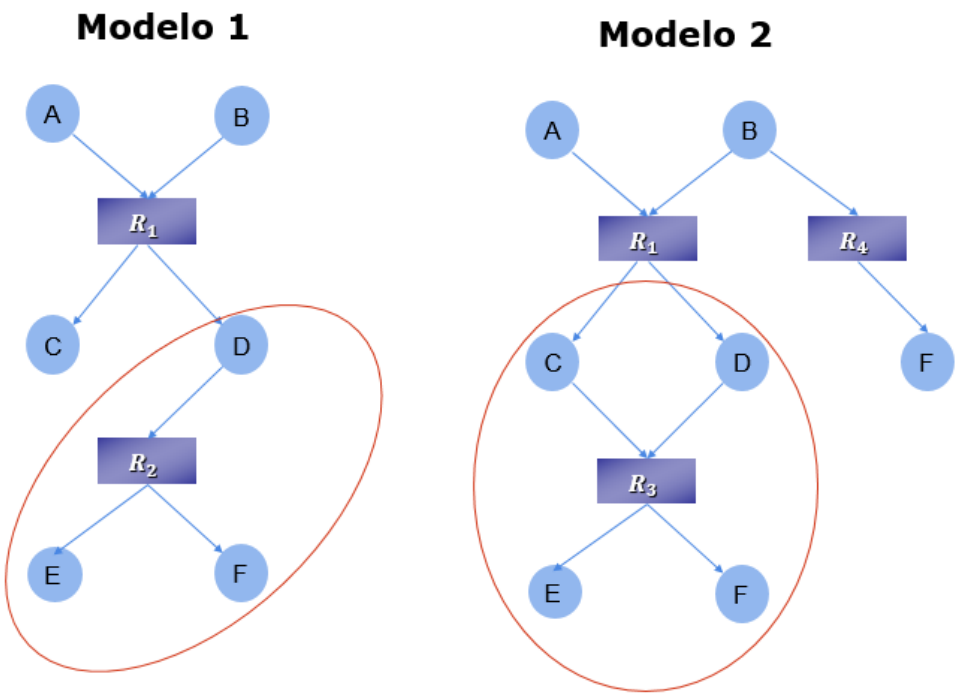


Figura 9: Dos reacciones diferentes que poseen cierta similitud en dos modelos diferentes.

5.3 Métricas para distancia entre dos modelos

Para las aplicaciones mencionadas en el epígrafe 5.1, es necesario tener una medida que indique qué tan parecidos o qué tan diferentes son dos modelos metabólicos, en particular los que tienen diferente orden (diferente número de metabolitos) y/o tamaño (diferente número de aristas).

Haciendo nuevamente una analogía entre grafos y modelos metabólicos, para cuantificar la similitud entre dos modelos debemos utilizar métricas. Recordemos que una métrica d sobre un conjunto X es una función $d: X \times X \to \mathbb{R}$ que cumple:

I) $d\ (x; y) \geq 0$,

II) $d\ (x; y) = 0$, si y sólo si $x = y$,

III) $d\ (x; y) = \ d\ (y; x)$,

IV) $d\ (x; y) + d\ (y; z)\ \geq d\ (z; y)$,

para todo $x, y, z\ \in X$ (Munkres et al. 2002).

A continuación, se muestran algunas de las métricas que se han propuesto, específicamente para comparar grafos moleculares y otras que pertenecen a otros

campos de conocimiento, las cuales, sin embargo, también podrían ser de utilidad al comparar modelos metabólicos. Aunque previamente es necesario exponer algunos conceptos básicos de la teoría de grafos, necesarios para entender las métricas propuestas.

- Un grafo G es un par $G_i = (V_i; E_i)$, donde V_i es su conjunto de vértices y E_i es su conjunto de aristas.
- El orden de un grafo es su número de vértices y su tamaño es su número de aristas.
- Sean $G_i = (V_i; E_i)$ y $G_j = (V_j; E_j)$ dos grafos:

  Un subgrafo de G_i es un grafo con todos sus vértices y aristas en G_i. El subgrafo común máximo (MCS(G_i; G_j)) de G_i; G_j es un subgrafo de G_i y de $G_j$con el mayor número posible de aristas (Baláž et al. 1989).

  El supergrafo común mínimo ($MCS(G_i;\ G_j)$) de G_i; G_j es un grafo $G_s = (V_s;\ E_s)$ = tal que $E_i \subseteq E_s$; $E_j \subseteq E_s$ y G_s tiene el número más pequeño posible de aristas (Baláž et al. 1989).

5.4.1 Métrica propuesta por Baláž

La función:

$$d_\alpha(G_i, G_j) = |E_i| + |E_j| - 2\left|E_{MCS(G_i,G_j)}\right| + \left||V_i| - |V_j|\right|,$$

Donde $|E_i|$ y $|E_j|$ representa el número de aristas en los grafos G_i y G_j respectivamente y $|V_i|$ y $|V_j|$el número de vértices de ambos grafos, fue propuesta en 1986 (Baláž *et al.* 1986).

Esta métrica cuenta las aristas del máximo subgrafo común en $E_{MCS(G_i,G_j)}$ que no están ni en E_i ni en E_j y el número de vértices que hacen que G_i y G_j sean de diferente orden.

En este caso tendríamos que unir todas las reacciones que aparecen en ambos modelos y contar las aristas que tiene cada una de las reacciones. En cada modelo por separado tendríamos que contar también las aristas y los metabolitos.

	Amycolatopsis balhimycina	iSyn811	Saccharomyce scerevisiae	Streptomyces coelicolor	iSyf715
Amycolatopsis balhimycina	0	5970	6178	5775	5733
iSyn811		0	7054	5401	417
Saccharomyces cerevisiae			0	5275	6917
Streptomyces coelicolor				0	5582
iSyf715					0

Tabla 3: Comparaciones utilizando las métrica de Baláž entre los modelos Amycolatopsis balhimycina, iSyn811, Saccharomyces cerevisiae, Streptomyces coelicolor y la iSyf715.

5.4.2 Métrica propuesta por Bunke

Sean $G_i = (V_i, E_i)$ *y* $G_j = (V_j, E_j)$, dos grafos, la distancia entre los dos está dada por:

$$d(G_i, G_j) = 1 - \frac{|MCS(G_i, G_j)|}{\max(|G_i|, |G_j|)}$$

donde $|G_i|$ y $|G_j|$ son el número de vértices de los grafos G_i y G_j respectivamente (Bunke et al. 1998). Si se usa en el denominador la expresión $|G_i| + |G_j| - |G_{ij}|$ en vez de max($|G_i|, |G_j|$), la cual corresponde al número de vértices de $G_i \cup G_j$, se obtiene otra métrica (Wallis et al. 2001). Si se usa el número de aristas en lugar del número de vértices, se sigue obteniendo una métrica (Wallis et al. 2001).

Al aplicar la métrica de Bunke en modelos metabólicos se toman como $MCS(G_i, G_j)$ los metabolitos que aparecen en ambos modelos y como $\max(|G_i|, |G_j|)$ se toma el que más metabolitos tenga entre G_i y G_j.

	Amycolatopsis balhimycina	iSyn811	Saccharomyces.cerevisiae	Streptomyces coelicolor	iSyf715
Amycolatopsis balhimycina	0	0.973	0.900	0.971	0.973
iSyn811		0	0.953	0.969	0.188
Saccharomyces.cerevisiae			0	0.898	0.96
Streptomyces coelicolor				0	0.968
iSyf715					0

Tabla 4: Comparaciones utilizando las métrica de Bunke entre los modelos Amycolatopsis balhimycina, iSyn811, Saccharomyces cerevisiae, Streptomyces coelicolor y la iSyf715.

5.4.3 Distancia de Edición

H. Bunke (Bunke *et al. 1997)* definió formalmente la distancia de edición en 1997. Sea edición una función de coste, sea e_i una operación de edición (inserción, borrado o cambio de nodos) y sea $\gamma(G_1, G_2)$ un conjunto de procesos para transformar el grafo G_1 en el grafo G_2, la función:

$$d_{ED}(G_1, G_2) = \min_{(e_1,\ldots\ldots,e_k)\in\gamma(G_1,G_2)} \sum_{i=1}^{k} \big(Edición(e_i)\big),$$

es una métrica, si $d_{ED} > 0$ para todo par de grafos que no sean iguales, 0 si son iguales, y si d_{ED} entre $G_1\ y\ G_2$ sea igual a d_{ED} entre $G_2\ y\ G_1$(Munkres et al. 2002).

En esta métrica se contabilizan todas las modificaciones que se hacen en un modelo para convertirlo en otro, por ejemplo, adicionar o eliminar un metabolito o incluir un metabolito que existe en una reacción que se crea o modifica. Los cambios de edición que se contabilizan son a nivel de metabolitos. En un modelo se puede crear una reacción desde cero y lo que contabiliza el sistema es: los nuevos metabolitos que se incluyen o se eliminan en el modelo (como adición o eliminación de nodos) más la

cantidad de metabolitos que están ya dentro del modelo que se adicionan o se restan a la reacción.

	Amycolatopsis balhimycina	iSyn811	Saccharomyces.cerevisiae	Streptomyces coelicolor	iSyf715
Amycolatopsis balhimycina	0	1778	1636	652	1027
iSyn811		0	1073	741	161
Saccharomyces.cerevisiae			0	661	1010
Streptomyces coelicolor				0	1022
iSyf715					0

Tabla 5: Comparaciones utilizando las métrica de Distancia de Edición entre los modelos Amycolatopsis balhimycina, iSyn811, Saccharomyces cerevisiae, Streptomyces coelicolor y la iSyf715.

5.5 Descripción del Software

La herramienta CompNet es una herramienta desarrollada en Visual Studio, específicamente en el lenguaje de programación C# y que sus principales clases son: “TKEGGReactionsProvider”, “TMetabolite”, “TReaction”, “TReactionsGraph”, “ModifyNameMetabolites”, “DistanceGraphCalculator” y “ModificationsofGraph”, con una interfaz visual que permite al usuario interactuar con la herramienta sin la necesidad de utilizar comandos como se muestra en la figura 5.4. Es una herramienta que permite obtener, a partir de la información de dos modelos metabólicos, las semejanzas y diferencias entre ambos modelos. Ofrece información de qué tan diferentes o iguales son dos modelos a partir del cálculo de varias métricas y, en caso de que se desee realizar algún tipo de modificación para que un modelo tenga características similares al otro, sugiere las posibles modificaciones.

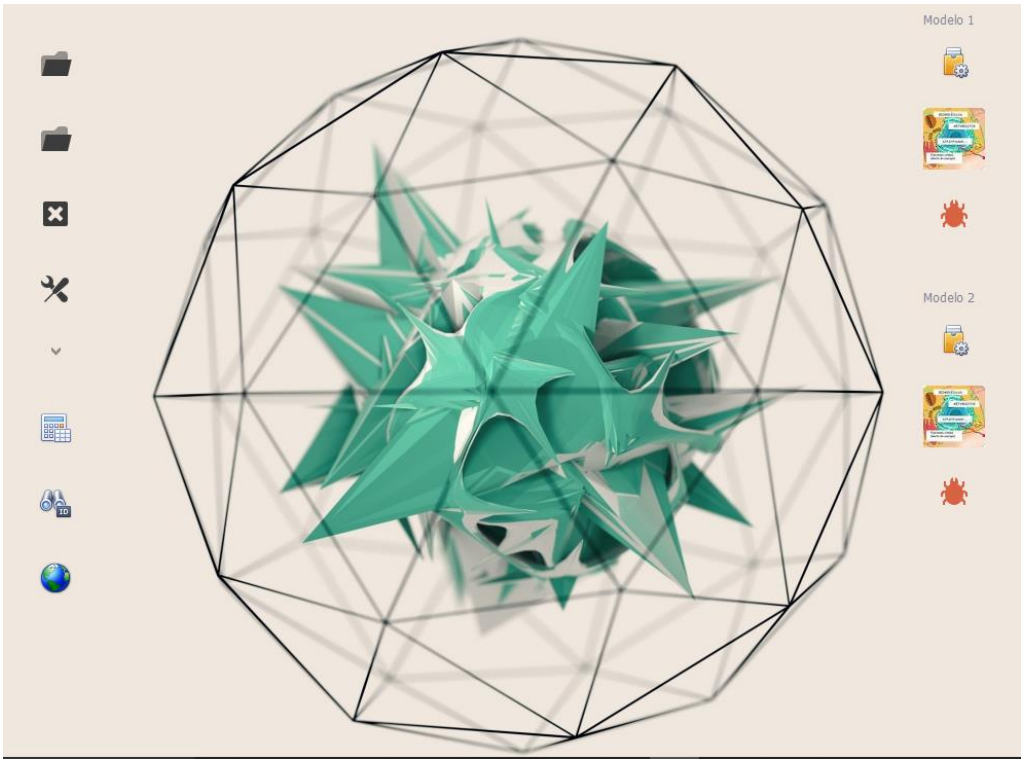


Figura 10: Interfaz principal de la aplicación CompNet.

La clase “TKEGGReactionsProvider” depende de las clases “TMetabolite” y “TReaction”. Esta clase es la encargada de realizar la entrada de la información de los dos modelos a comparar que pueden ser obtenidos a partir de ficheros OptGene o sbml. Una vez que lee los ficheros, va construyendo los objetos “TMetabolite” y “TReaction”. De estas dos clases depende la clase “TReactionsGraph”, pues a medida que se van construyendo las “TReaction” con sus “TMetabolite” se van incluyendo en un objeto de la clase “TReactionsGraph” para formar dos grafos que representen a los modelos metabólicos a comparar.

Después de tener guardados los dos modelos metabólicos en dos objetos de tipo “TReactionsGraph”, necesitamos que ambos estén con la misma nomenclatura y para ello dependemos de la clase “ModifyNameMetabolites”, la cual tomará un modelo y le modificará los nombres de los metabolitos en función del otro modelo a partir de los alias que tiene el nombre de cada metabolito en KEGG. Es importante señalar que, en ambos modelos, es necesario que se utilicen para cada metabolito alguno de los alias que propone KEGG. Una vez que se tengan en ambos objetos los mismos nombres para cada metabolito, podemos proceder a compararlos.

En la clase “DistanceGraphCalculator” implementamos las métricas antes mencionadas en los epígrafes 5.4.1, 5.4.2 y 5.4.3. Para tener una idea de lo diferentes que pueden ser los modelos, en la clase “ModificationsofGraph” se pueden registrar los cambios que tendría que llevarse a cabo en un modelo para convertirlo en el otro.

5.5.1 Clase TKEGGReactionsProvider

Esta clase está hecha para tomar la información de los modelos que se utilizarán como orígenes de datos, por lo que tiene métodos simples para el trabajo con ficheros como son “start()” y “readReaction()”. Posee el método “GetReaction” al que se le pasa como parámetro un string que es una línea del fichero (en caso de que el origen de datos sea OptGene). Con esta línea se procede a descartar que sea un comentario en el fichero, para después comprobar que sea una reacción que puede ser reversible “<->” o irreversible “->” y dividirla en reaccionantes y productos, los cuales se separan por el signo “+” y se crean como objetos “TMetabolite” que se guardan entre los reaccionantes o productos de esa reacción. En caso de que fuera un fichero con el formato sbml, se buscarían las reacciones para ir construyéndolas una a una de la misma manera. A medida que se recorre el fichero se van creando y adicionando los metabolitos con las reacciones.

5.5.2 Clase TMetabolite

Esta clase es la encargada de gestionar la información de los metabolitos del modelo metabólico mediante campos como: “_reactionIn” que es un objeto de tipo diccionario donde se guardan las reacciones en las que aparece un metabolito determinado, así como “name” para guardar el nombre del metabolito y “native_reaction” para determinar posteriormente si es un metabolito nativo de la reacción o uno que se incluye cuando modificamos el modelo.

5.5.3 Clase TReaction

La clase “TReaction” es la encargada de construir las reacciones con todos los campos y funcionalidades necesarias para poder comparar las reacciones de cada modelo. Posee campos para el id de la reacción, la reversibilidad, para saber si es una reacción nativa o modificada del modelo, un campo para saber el porciento de similitud que tiene con otra reacción del otro modelo. Lo más importante que posee es un diccionario de los metabolitos reaccionantes y de los productos que componen la reacción, así como dos diccionarios para guardar las modificaciones que se realicen entre los reaccionantes y los productos.

5.5.4 Clase TReactionsGraph

La clase “TReactionsGraph” es en la que se crea y almacena la información de los modelos metabólicos (los grafos a comparar). Para ello, dispone un diccionario para las reacciones que tiene el modelo, otro para las reacciones que se modifiquen y otro para los metabolitos que se adicionen o se eliminen del modelo. También posee un método llamado “includeReaction” al que se le pasa como parámetro una reacción que se comprueba si existe o no para adicionarla. La comprobación se hace primeramente por su id y, si no hay coincidencias, se comprueban por los componentes que forman la reacción una vez cargados los modelos en dos objetos “TReactionsGraph”.

5.5.5 Clase ModifyNameMetabolites

En esta clase, una vez que se tienen guardados los dos modelos metabólicos en dos objetos de tipo “TReactionsGraph”, es necesario que ambos modelos tengan el mismo nombre para el mismo metabolito. Por ejemplo, en un modelo, un metabolito puede llamarse Sodio y en otro Na, cuando en realidad estamos hablando del mismo compuesto. Por lo que se hace necesario, primero, que los metabolitos de cada modelo estén declarados con la notación de KEGG y, segundo, hacer que en ambos modelos la notación de los metabolitos sea la misma.

Para ello utilizamos un fichero en XML llamado “metabolito.xml” donde tenemos guardados el nombre más usado por cada metabolito y una lista de sus alias. De esta manera buscamos cada metabolito en los dos modelos y modificamos los nombres de

los metabolitos en uno de los modelos para que ambos tengan los mismos metabolitos con los mismos nombres.

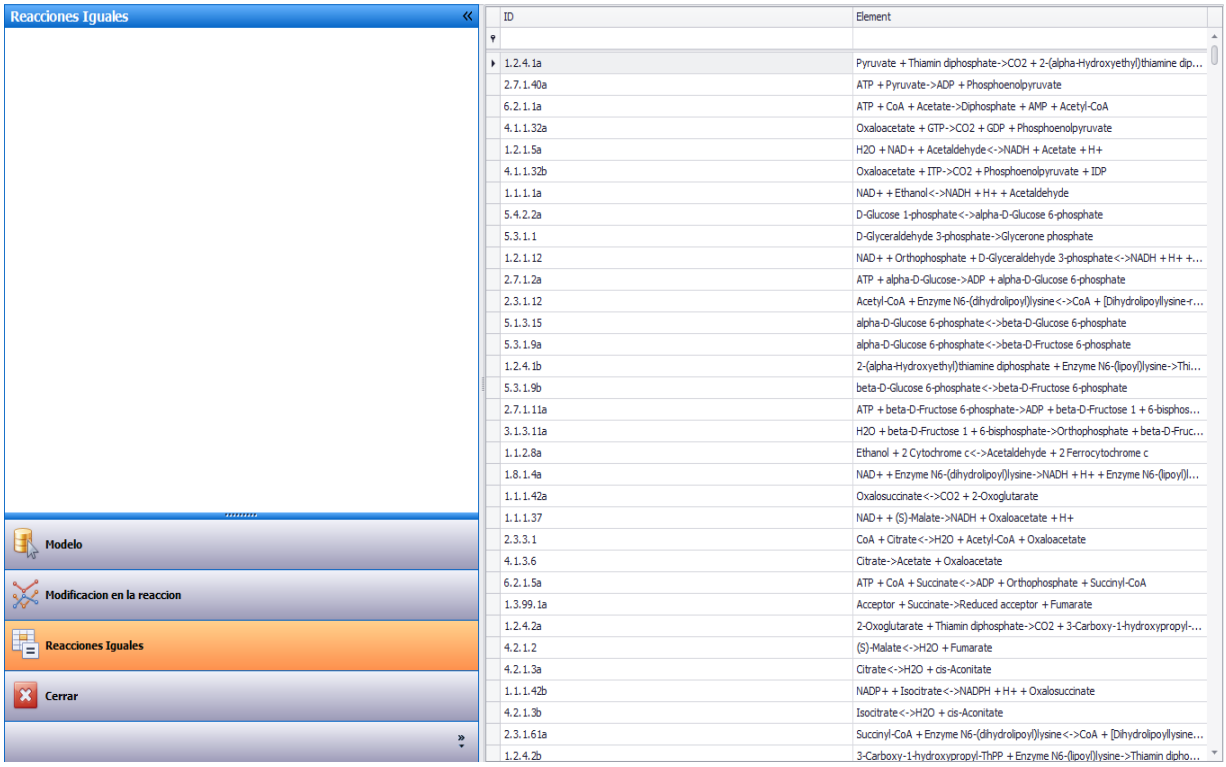
5.5.6 Clase DistanceGraphCalculator

La clase “DistanceGraphCalculator” como atributos tiene dos objetos de la clase “TReactionsGraph” que tiene la información de los dos modelos que se desea comparar. En esta clase se implementan los métodos que retornan un número que significa la distancia que hay entre un modelo y otro. Estos métodos están programados en función de las diferentes métricas mencionadas en los epígrafes 5.4.1, 5.4.2 y 5.4.3.

5.5.7 Clase ModificationsofGraph

En esta clase guardamos todos los cambios que se deben realizar en un modelo para que sea como el otro. Posee como atributos un objeto “TReactionsGraph” y un objeto de tipo diccionario que contiene string y objetos TReaction.

En el objeto “TReactionsGraph” que tiene esta clase se guardan primeramente las reacciones que son comunes para ambos modelos y, por ende, los metabolitos que coinciden en los dos modelos como se muestra en la figura 5.5. Después, se van adicionando o eliminando las reacciones que se modificaron con los cambios hechos.



ID	Element
1.2.4.1a	Pyruvate + Thiamin diphosphate->CO2 + 2-(alpha-Hydroxyethyl)thiamine dip...
2.7.1.40a	ATP + Pyruvate->ADP + Phosphoenolpyruvate
6.2.1.1a	ATP + CoA + Acetate->Diphosphate + AMP + Acetyl-CoA
4.1.1.32a	Oxaloacetate + GTP->CO2 + GDP + Phosphoenolpyruvate
1.2.1.5a	H2O + NAD+ + Acetaldehyde<->NADH + Acetate + H+
4.1.1.32b	Oxaloacetate + ITP->CO2 + Phosphoenolpyruvate + IDP
1.1.1.1a	NAD+ + Ethanol<->NADH + H+ + Acetaldehyde
5.4.2.2a	D-Glucose 1-phosphate<->alpha-D-Glucose 6-phosphate
5.3.1.1	D-Glyceraldehyde 3-phosphate->Glycerone phosphate
1.2.1.12	NAD+ + Orthophosphate + D-Glyceraldehyde 3-phosphate<->NADH + H+ +...
2.7.1.2a	ATP + alpha-D-Glucose->ADP + alpha-D-Glucose 6-phosphate
2.3.1.12	Acetyl-CoA + Enzyme N6-(dihydrolipoyl)lysine<->CoA + [Dihydrolipoyllysine-r...
5.1.3.15	alpha-D-Glucose 6-phosphate<->beta-D-Glucose 6-phosphate
5.3.1.9a	alpha-D-Glucose 6-phosphate<->beta-D-Fructose 6-phosphate
1.2.4.1b	2-(alpha-Hydroxyethyl)thiamine diphosphate + Enzyme N6-(lipoyl)lysine->Thi...
5.3.1.9b	beta-D-Glucose 6-phosphate<->beta-D-Fructose 6-phosphate
2.7.1.11a	ATP + beta-D-Fructose 6-phosphate->ADP + beta-D-Fructose 1 + 6-bisphos...
3.1.3.11a	H2O + beta-D-Fructose 1 + 6-bisphosphate->Orthophosphate + beta-D-Fruc...
1.1.2.8a	Ethanol + 2 Cytochrome c<->Acetaldehyde + 2 Ferrocytochrome c
1.8.1.4a	NAD+ + Enzyme N6-(dihydrolipoyl)lysine->NADH + H+ + Enzyme N6-(lipoyl)l...
1.1.1.42a	Oxalosuccinate<->CO2 + 2-Oxoglutarate
1.1.1.37	NAD+ + (S)-Malate->NADH + Oxaloacetate + H+
2.3.3.1	CoA + Citrate<->H2O + Acetyl-CoA + Oxaloacetate
4.1.3.6	Citrate->Acetate + Oxaloacetate
6.2.1.5a	ATP + CoA + Succinate<->ADP + Orthophosphate + Succinyl-CoA
1.3.99.1a	Acceptor + Succinate->Reduced acceptor + Fumarate
1.2.4.2a	2-Oxoglutarate + Thiamin diphosphate->CO2 + 3-Carboxy-1-hydroxypropyl-...
4.2.1.2	(S)-Malate<->H2O + Fumarate
4.2.1.3a	Citrate<->H2O + cis-Aconitate
1.1.1.42b	NADP+ + Isocitrate<->NADPH + H+ + Oxalosuccinate
4.2.1.3b	Isocitrate<->H2O + cis-Aconitate
2.3.1.61a	Succinyl-CoA + Enzyme N6-(dihydrolipoyl)lysine<->CoA + [Dihydrolipoyllysine...
1.2.4.2b	3-Carboxy-1-hydroxypropyl-ThPP + Enzyme N6-(lipoyl)lysine->Thiamin dipho...

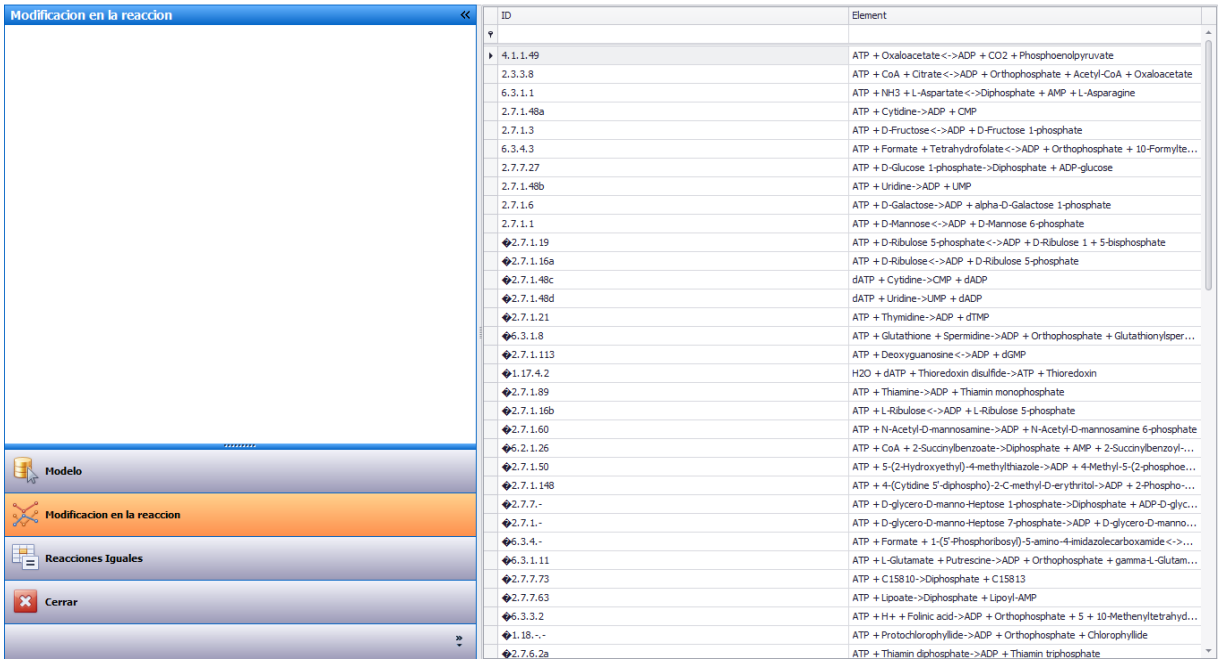
Figura 11: Reacciones que coinciden en dos modelos.

El método “RE_Wedding” toma de los dos modelos las reacciones que no son comunes y cada reacción del primer modelo la empareja con la reacción del otro modelo que mejor porciento de igualdad tiene. Cada reacción de un modelo que no se encuentra en el otro tiene calculado un ranking de las reacciones que son diferentes en el otro modelo- ordenado de mayor a menor por el porciento de igualdad que tenga con las reacciones del otro modelo. Si hay dos reacciones en un modelo que tienen el mismo porciento de igualdad con una reacción del otro modelo, cada una de las reacciones del primer modelo busca las siguientes reacciones que le siguen en su ranking y se queda con la que mejor porciento de igualdad tiene en segunda opción.

Después de que el método “RE_Wedding” decide cómo se van a combinar las reacciones de uno y otro modelo, utilizamos dos métodos llamados “Match_Products” y “Match_Reactants” que son los que registran las modificaciones que se deben realizar en los reaccionantes y los productos de una reacción para que sea igual a la otra.

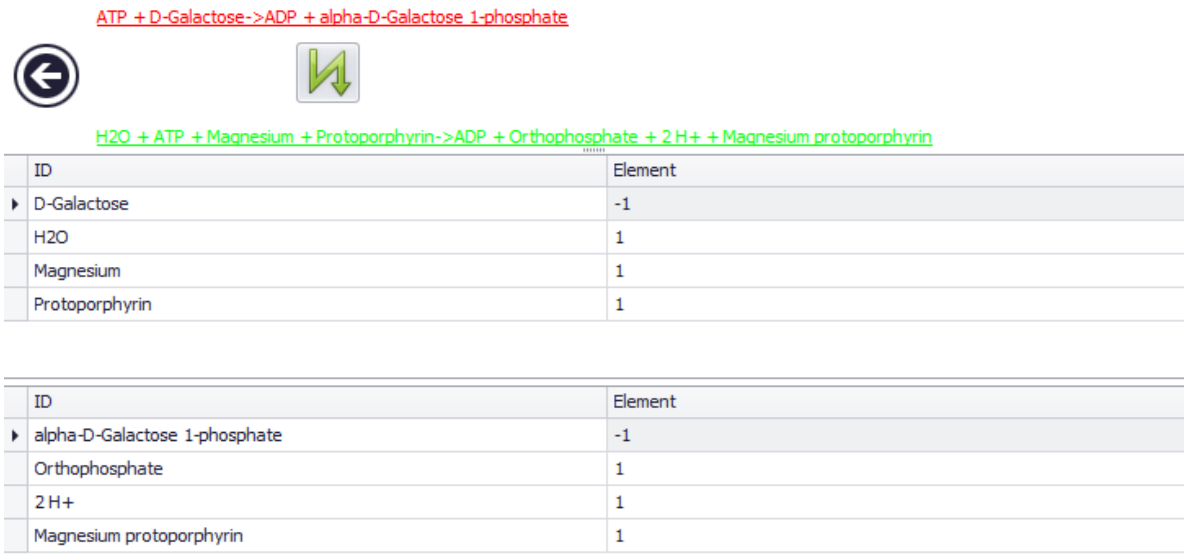
Con todos estos movimientos, además de registrar los metabolitos que se incluyen o excluyen de una reacción, se registra cuándo el metabolito entra o sale del modelo.

Como se muestra en la figura 5.6, se pueden ver las reacciones que fueron modificadas en uno de los dos modelos metabólicos y, si seleccionamos la reacción, podemos ver las modificaciones que se le realizaron (como muestra la figura 5.7).



ID	Element
4.1.1.49	ATP + Oxaloacetate<->ADP + CO2 + Phosphoenolpyruvate
2.3.3.8	ATP + CoA + Citrate<->ADP + Orthophosphate + Acetyl-CoA + Oxaloacetate
6.3.1.1	ATP + NH3 + L-Aspartate<->Diphosphate + AMP + L-Asparagine
2.7.1.48a	ATP + Cytidine->ADP + CMP
2.7.1.3	ATP + D-Fructose<->ADP + D-Fructose 1-phosphate
6.3.4.3	ATP + Formate + Tetrahydrofolate<->ADP + Orthophosphate + 10-Formylte...
2.7.7.27	ATP + D-Glucose 1-phosphate->Diphosphate + ADP-glucose
2.7.1.48b	ATP + Uridine->ADP + UMP
2.7.1.6	ATP + D-Galactose->ADP + alpha-D-Galactose 1-phosphate
2.7.1.1	ATP + D-Mannose<->ADP + D-Mannose 6-phosphate
2.7.1.19	ATP + D-Ribulose 5-phosphate<->ADP + D-Ribulose 1 + 5-bisphosphate
2.7.1.16a	ATP + D-Ribulose<->ADP + D-Ribulose 5-phosphate
2.7.1.48c	dATP + Cytidine->CMP + dADP
2.7.1.48d	dATP + Uridine->UMP + dADP
2.7.1.21	ATP + Thymidine->ADP + dTMP
6.3.1.8	ATP + Glutathione + Spermidine->ADP + Orthophosphate + Glutathionylsper...
2.7.1.113	ATP + Deoxyguanosine<->ADP + dGMP
1.17.4.2	H2O + dATP + Thioredoxin disulfide->ATP + Thioredoxin
2.7.1.89	ATP + Thiamine->ADP + Thiamin monophosphate
2.7.1.16b	ATP + L-Ribulose<->ADP + L-Ribulose 5-phosphate
2.7.1.60	ATP + N-Acetyl-D-mannosamine->ADP + N-Acetyl-D-mannosamine 6-phosphate
6.2.1.26	ATP + CoA + 2-Succinylbenzoate->Diphosphate + AMP + 2-Succinylbenzoyl-...
2.7.1.50	ATP + 5-(2-Hydroxyethyl)-4-methylthiazole->ADP + 4-Methyl-5-(2-phosphoe...
2.7.1.148	ATP + 4-(Cytidine 5'-diphospho)-2-C-methyl-D-erythritol->ADP + 2-Phospho-...
2.7.7.-	ATP + D-glycero-D-manno-Heptose 1-phosphate->Diphosphate + ADP-D-glyc...
2.7.1.-	ATP + D-glycero-D-manno-Heptose 7-phosphate->ADP + D-glycero-D-manno...
6.3.4.-	ATP + Formate + 1-(5'-Phosphoribosyl)-5-amino-4-imidazolecarboxamide<->...
6.3.1.11	ATP + L-Glutamate + Putrescine->ADP + Orthophosphate + gamma-L-Glutam...
2.7.7.73	ATP + C15810->Diphosphate + C15813
2.7.7.63	ATP + Lipoate->Diphosphate + Lipoyl-AMP
6.3.3.2	ATP + H+ + Folinic acid->ADP + Orthophosphate + 5 + 10-Methenyltetrahyd...
1.18.-.-	ATP + Protochlorophyllide->ADP + Orthophosphate + Chlorophyllide
2.7.6.2a	ATP + Thiamin diphosphate->ADP + Thiamin triphosphate

Figura 12: Reacciones que fueron modificadas.



ID	Element
D-Galactose	-1
H2O	1
Magnesium	1
Protoporphyrin	1

ID	Element
alpha-D-Galactose 1-phosphate	-1
Orthophosphate	1
2 H+	1
Magnesium protoporphyrin	1

Figura 13: Listado de modificaciones en una reacción.

Capítulo 6. Conclusiones

En el presente trabajo se desarrolló un grupo de métodos y herramientas de simulación, aplicado a la optimización de funciones objetivo biológicas para el análisis de modelos metabólicos. Esta investigación se ha basado en métodos no tradicionales que permiten ofrecer mejoras en los resultados de las simulaciones, con un mejor acercamiento a la realidad en el contexto de la ingeniería metabólica.

La complejidad de los modelos y las redes metabólicas a escala genómica, asociados a sistemas biológicos, hace que el uso de las herramientas computacionales sea un elemento esencial en el campo de la biología de sistemas.

En este sentido se presentó **PyNetMet**, una librería de Python, como herramienta para trabajar con redes y modelos metabólicos. Con el fin de ilustrar las características más importantes y algunos de sus usos, se mostraron resultados de la herramienta como el cálculo de la agrupación media de las redes que representan a cada uno de los modelos metabólicos, el número de metabolitos desconectados en cada modelo y la distancia media entre dos metabolitos cualesquiera de la red. Además, la aplicación permite calcular los parámetros de la red para la versión no dirigida de la misma, obtenidos a partir de la conexión de los metabolitos que aparecen como sustratos y productos en las reacciones del modelo.

A menudo, analizar los modelos metabólicos partiendo de la optimización monobjetivo no siempre se acerca todo lo deseado a la realidad, puesto que uno o más objetivos pueden entrar en conflicto porque tienen como denominador común la necesidad de elegir entre diferentes alternativas que han de evaluarse en base a diversos criterios. Para ello, se presentó un algoritmo de optimización multiobjetivo basado en algoritmos evolutivos que consiste en una adaptación del algoritmo sp-MODE (Reynoso-Meza et al., 2010, Reynoso-Meza et al., 2014).

Se validó una herramienta bioinformática llamada **BioMOE**, basada en el algoritmo sp-MODE, que considera de manera simultánea la optimización de dos o más objetivos, a menudo en conflicto, dando como soluciones diferentes distribuciones de flujo en la que una no es mejor que la otra.

En el área de la comparación de modelos metabólicos se desarrolló una herramienta bioinformática llamada CompNet, basada en conceptos de teoría de grafos como las Redes de Petri, para poder establecer una comparación entre modelos metabólicos, determinando qué cambios serían necesarios para modificar determinadas funciones en uno de los modelos con respecto al otro, a través de la métrica Distancia de Edición.

Se ha demostrado, mediante las métricas de Baláž y Bunke, la similitud entre los modelos Amycolatopsis balhimycina, iSyn811, Saccharomyces cerevisiae, Streptomyces coelicolor y iSyf715, mostrándose el grado de semejanza que existe entre ellos. La herramienta permite obtener un valor cuantitativo que indica las semejanzas y diferencias que hay entre dos modelos, a partir del uso de diferentes métricas, demostrándose el grado de semejanza que poseen los modelos iSyn811 y iSyf715.

En este trabajo se ejecutaron las pruebas de validación necesarias para mostrar que los métodos y herramientas cumplen la funcionalidad demandada y que ofrecen una mejor comprensión de las redes metabólicas.

Bibliografía

Palsson BØ. (2006) **Systems Biology - Properties of Reconstructed Networks,** *Cambridge University Press: New York.*

Bard J. (2013) **Systems biology-the broader perspective. Cells, 2:414-431.**

Klipp E, Herwig R, Kowald A, Wierling C, Lehrach H, *et al.* (2005) **Systems Biology in Practice – Klipp.**

Snoep JL, Bruggeman F, Olivier BG. *et al.* (2006) **Towards building the silicon cell: A modular approach**. *BioSystems,* **83**:207-216.

López M, Ruiz G, Vega M, *et al.* (2007) **Informe de vigilancia tecnológica. Edición: Cintia Refojo,** Genoma España. Referencia: GEN-ES07003**, ISBN: 84-609-9762-6.**

Förster J, Famili I, Fu P, *et al.* (2003) **Genome-scale reconstruction of the *Saccharomyces cerevisiae* metabolic network.** Genome Res**, 13:**244-253**.**

O'Brien EJ, Lerman JA, Chang RL, *et al.* (2013) **Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction**. *Mol Syst Biol*, **9**:693.

Montagud A, Navarro E, Fernández de Córdoba P, *et al.* (2010) **Reconstruction and analysis of genome-scale metabolic model of a photosynthetic bacterium**. *BMC Syst Biol,* **4**:156-172.

Karr JR, Sanghvi JC, Macklin DN, *et al.* (2012) **A whole-cell computational model predicts phenotype from genotype**. *Cell* **150**:389-401.

Esvelt KM, Wang HH, *et al.* (2013) **Genome-scale engineering for systems and synthetic biology**. *Mol Syst Biol*, **9(641)**:1-17.

Baart GJE, Martens DE, *et al.* (2012) **Genome-scale metabolic models: reconstruction and analysis**. *Methods Mol Biol*, **799**:107-126.

Tomita M. (2001) **Whole-cell simulation: a grand challenge of the 21st century**. *Trends Biotechnol*, **19**:205-210.

Covert MW, Knight EM, Reed JL, *et al.* (2004) **Integrating high-throughput and computational data elucidates bacterial networks**. *Nature*, **429**:92-96.

Bruggeman FJ, Westerhoff HV, *et al.* (2006) **Approaches to biosimulation of cellular processes**. *J Biol Phys*, **32**:273-288.

Bruggeman FJ, Snoep JL, Westerhoff HV, *et al.* (2008) **Control, responses and modularity of cellular regulatory networks: a control analysis perspective**. *IET Syst Biol*, **2**:397-410.

De Mey M, Taymaz-Nikerel H, Baart G, *et al.* (2010) **Catching prompt metabolite dynamics in *Escherichia coli* with the BioScope at oxygen rich conditions**. *Metab Eng*, **12**:477-487.

Taymaz-Nikerel H, Borujeni AE, Verheijen PJ, *et al.* (2010) **Genome-derived minimal metabolic models for *Escherichia coli* MG1655 with estimated in vivo respiratory ATP stoichiometry**. *Biotechnol Bioeng*, **107**:369-381.

Varma A, Palsson BØ. *et al.* (1993) **Metabolic capabilities of E coli II. Optimal growth patterns**. *J Theor Biol*, **165**:503-522.

Edwards JS, Ramakrishna R, Schilling CH, Palsson BØ, *et al.* (1999) **Metabolic flux balance analysis.** In *Metabolic Engineering*. Edited by Lee S, Papoutsakis E New York: Marcel Dekker Inc.

Schilling CH, Schuster S, Palsson BØ, *et al.* (1999) **Metabolic pathway analysis: basic concepts and scientific applications in the postgenomic era**. *Biotechnol Prog*, **15**:296-303.

Varma A, Palsson BØ, *et al.* (1994c) **Stoichiometric flux balance models quantitatively predict growth and metabolic by-product excretion in wild-type *Escherichia coli* W3110**. *Appl Environ Microb,* **60**:3724-3731.

Schuster S, Dandekar T, Fell DA, *et al.* (1999) **Detection of elementary flux modes in biochemical networks: a promising tool for pathway analysis and metabolic engineering**. *Trends Biotechnol*, **17**:53-60.

Edwards JS, Palsson BØ, *et al.* (2000) **Metabolic flux balance analysis and the *in silico* analysis of *Escherichia coli* K-12 gene deletions**. *BMC Bioinformatics*, **1**:1-10.

Edwards JS, Ramakrishna R, Palsson BØ, *et al.* (2002) **Characterizing the Metabolic Phenotype: a Phenotype Phase Plane Analysis**. *Biotechnol Bioeng*, **77**:27-36.

Segrè D, Vitkup D, Church GM, *et al.* (2002) **Analysis of optimality in natural and perturbed metabolic networks**. *PNAS*, **99**:15112-15117.

Mahadevan R, Schilling CH, *et al.* (2003) **The effects of alternate optimal solutions in constraint-based genome-scale metabolic models**. *Metab Eng*, **5**:264-276.

Shlomi T, Berkman O, Ruppin E, *et al.* (2005) **Regulatory on/off minimization of metabolic flux changes after genetic perturbations**. *PNAS*, **102**:7695-7700.

Price ND, Reed JL, Palsson BØ, *et al.* (2004b) **Genome-scale models of microbial cells: evaluating the consequences of constraints**. *Nat Rev Microbiology,* **2**:886-897.

Elowitz MB, Surette MG, Wolf PE, *et al.* (1999) **Protein mobility in the cytoplasm of *Escherichia coli***. *J Bacteriol,* **181**:197-203.

Werner A, Heinrich R, *et al.* (1985). **A kinetic model for the interaction of energy metabolism and osmotic states of human erythrocytes. Analysis of the stationary ''*in vivo*'' state and of time dependent variations under blood preservation conditions**. *Biomed Biochim Acta*, **44**:185-212.

Lew VL, Bookchin RM, *et al.* (1986) **Volume, pH, and ion-content regulation in human red cells: analysis of transient behavior with an integrated model**. *J Membr Biol*, **92**:57-74.

Price ND, Reed JL, Palsson BO, *et al.* (2004) **Genome-scale models of microbial cells: Evaluating the consequences of constraints.** Nature Reviews Microbiology**, 2:**886–897**.**

Reed JL, and Palsson BO, *et al*. (2003) **Thirteen years of building constraints-based in silico models of Escherichia coli.** Journal of Bacteriology**, 185:**2692–2699**.**

Cases I, de Lorenzo V, Ouzounis CA, *et al.* (2003) **Transcription regulation and environmental adaptation in bacteria.** Trends in Microbiology**, 11:**248–253**.**

Bazaraa MS, Sherali HD, and Shetty CM, *et al.* (1993) **Nonlinear Programming: Theory and Algorithms,** 2nd edition. John Wiley and Sons, NJ, Hoboken**.**

Kell D and Mendes P, *et al* (2000). **Snapshots of systems: metabolic control analysis and biotechnology in the postgenomic era. In: Cornish-Bowden AJ, Cardenas ML, (eds). Technical and Medical Implications of Metabolic Control Analysis.** Amsterdam: Kluwer**, 2–**25**.**

Teusink B, Passarge J, Reijenga CA, et al. (2000). **Can yeast glycolysis be understood in terms of in vitro kinetics of the constituent enzymes? Testing biochemistry.** Eur J Biochem**; 267:**5313–29**.**

Beard DA, Liang SD, Qian H, et al. (2002). **Energy balance for analysis of complex metabolic networks.** BiophysJ**; 83:**79–86**.**

Covert MW, Famili I, Palsson BO, et al. (2003). **Identifying constraints that govern cell behavior: a key to converting conceptual to computational models in biology?** Biotechnol Bioeng**; 84:** 763–72**.**

Price ND, Reed JL, Palsson BO, et al. (2004). **Genome-scale models of microbial cells: evaluating the consequences of constraints.** Nat RevMicrobiol**; 2:**886–97**.**

Segre D, Vitkup D, Church GM, et al. (2002). **Analysis of optimality in natural and perturbed metabolic networks.** ProcNatlAcad Sci USA**; 99:**15112–7.

Varma A, Palsson BO, et al. (1994). **Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110.** Appl Environ Microbiol**; 60:**3724–31**.**

Edwards JS, Palsson BO, et al. (2000). **The Escherichia coli MG1655 in silico metabolic genotype: its definition, characteristics, and capabilities.** ProcNatl Acad SciUSA**; 97:**5528–33**.**

Ramakrishna R, Edwards JS, McCulloch A, Palsson BO, et al. (2001). **Flux-balance analysis of mitochondrial energy metabolism: consequences of systemic stoichiometric constraints.** Am J Physiol Regul Integr Comp Physiol**; 280:**R695–704.

Varma A, Boesch BW, Palsson BO, et al. (1993). **Stoichiometric interpretation of Escherichia coli glucose catabolism under various oxygenation rates.** Appl Environ Microbiol**; 59:** 2465–73**.**

Hjersted JL, Henson MA, et al. (2009). **Steady-state and dynamic flux balance analysis of ethanol production by Saccharomyces cerevisiae.** IET Syst Biol **3(3):**167–179.

T Shlomi, O Berkman, E Ruppin, et al. (2005). **Regulatory on_off minimization of metabolic flux changes after genetic perturbationsUniversity of Washington School of Medicine,** Seattle, WA**.**

Reynoso-Meza G, *et al* (2014) **Controller Tuning by Means of Evolutionary Multiobjective Optimization: a Holistic Multiobjective Optimization Design Procedure**. Editorial Universitat Politécnica de Valencia, 2014.

Paula MP *et al*. 2014. **Adaptación de herramientas de optimización monoobjetivo y multiobjetivo aplicadas a problemas de simulación de sistemas biológicos.** http://hdl.handle.net/10251/59531.

Förster J, Famili, I.P Fu, Palsson B and Nielsen J *et al*. (2003). **Genome-scale reconstruction of the Saccharomyces cerevisiae metabolic network.** Genome Research **13:** 244-53**.**

Montagud A, Navarro E, F. de Córdoba P, Urchueguía J.F. and Raosaheb K. *et al*. (2010). **Reconstruction and analysis of genome-scale metabolic model of a photosynthetic bacterium.** Systems biology 1752-0509/4/156**.**

Fang K, Zhao H, Sun Ch, Lam C M, Chang S, Zhang K, Panda G, Godinho M, Martins dos Santos V and Wang J *et al*. (2011). **Exploring the metabolic network of the epidemic pathogen Burkholderia cenocepacia J2315 via genome-scale reconstruction.** Systems biology 1752-0509/5/83**.**

Förster J, Famili IP Fu, Palsson B and Nielsen, *et al*. (2003). **Genome-scale reconstruction of the Saccharomyces cerevisiae metabolic network. Genome Research 13: 244-53.**

Imam S, Yilmaz S, Sohmen U, Gorzalski AS, Reed JL, Noguera DR and Donohue TJ, *et al.* (2011). **iRsp1095: A genome-scale reconstruction of the Rhodobacter sphaeroides metabolic network.** BMC Systems Biology: 116**.**

Milne CB, Eddy JA, Raju R, Ardekani S, Kim PJ, Senger RS, Jin YS, Blaschek HP and Price ND *et al*. (2011). **Metabolic network reconstruction and genome scale model of butanol-producing strain Clostridium beijerinckii NCIMB 8052.** BMC Systems Biology, 5:130, doi:**10.1186**/1752-0509-5-130.

Karp PD, Paley S and Romero P *et al.* (2002). **The Pathway Tools software. Bioinformatics 2002,** 18 Suppl **1:**S225-32**.**

Latendresse M, Krummenacker M, Trupp M, and Karp PD *et al.* (2012). **Construction and completion of flux balance models from pathway databases.** Bioinformatics Vol. 28 no. 3 2012, pages 388–396. doi:10.1093/bioinformatics/btr681)**.**

Sun J and Zeng AP *et al.* (2004). **IdentiCS – Identification of coding sequence and in silico reconstruction of the metabolic network directly from unannotated low-coverage bacterial genome sequence BMC Bioinformatics,** BMC Bioinformatics 2004, 5:112 doi:10.1186/1471-2105-5-112).

Pinney JW, Shirley MW, McConkey GA and Westhead DR *et al.* (2005). **metaSHARK: software for automated metabolic network prediction from DNA sequence and its application to the genomes of Plasmodium falciparum and Eimeria tenella.** Nucleic Acids Research, Vol. 33, No. 4 1399–1409. doi:10.1093/nar/gki285

Notebaart RA, van Enckevort F, Francke Ch, Siezen RJ and Teusink B *et al.* (2006) **Accelerating the reconstruction of genomescale metabolic networks,** BMC Bioinformatics. 7:296 doi:10.1186/1471-2105-7-296

Pitkänen E, Akerlund A, Rantanen A, Jouhten P and Ukkonen E *et al.* (2008). **ReMatch: a web-based tool to construct, store and share stoichiometric metabolic models with carbon maps for metabolic flux analysis.** J Integr Bioinform. 5(2). doi: 10.2390/biecoll-jib-2008-102.

Henry C, DeJongh M, Best A, Frybarger P, Linsay B and Stevens R *et al*. (2010). **High-throughput generation, optimization and analysis of genome-scale metabolic models.** Nature Biotechnology.**;28(9):**977–982. doi: 10.1038/nbt.1672**.**

Boele J, Olivier BG and Teusink B *et al*. (2012). **FAME, the Flux Analysis and Modeling Environment.** BMC System Biology.**6(8) doi:** 10.1186/1752-0509-6-8.

Liao YC, Chen JC, Tsai MH, Tang YH, Chen FC and Hsiung CA *et al.* (2011). **MrBac: a web server for draft metabolic network reconstructions for bacteria.** Bioeng Bugs. **Sep-Oct; 2(5):284-7.**

Feng X, Xu Y, Chen Y and Tang YJ *et al.* (2012). **MicrobesFlux: a web platform for drafting metabolic models from the KEGG database BMC.** Syst Biol**. 6: 94.** doi: 10.1186/1752-0509-6-94**.**

Liao YC, Tsai MH, Chen FC and Hsiung CA *et al.* (2012). **GEMSiRV: a software platform for GEnome-scale metabolic model simulation, reconstruction and visualization.** Bioinformatics**. 28(13):**1752-8.

Garrido J, Villar L, Reyes R, Jaime R, Triana J, Córdova V, Castro JC, Navarro E, Montagud A, F de Córdoba P, Urchueguía JF and Martínez J *et al.* (2011). **HYDRA: una plataforma informática orientada al diseño, análisis y visualización de redes metabólicas. XIV Convención y Feria Internacional Informática 2011.**La Habana, Cuba, del 7 al 11 de febrero de 2011. Publicado en: Programa Científico. Pág. 55**. ISBN 978-959-7213-01-7.**

Reyes R, Jaime R, Garrido J, Triana J, Villar L, Córdova V, Castro JC, Navarro E, Montagud A, F de Córdoba P, Urchueguía JF and Martínez J *et al*. (2011). **Base de datos biológica orientada a la automatización del proceso de construcción de modelos a escala genómica. Resultados en la Synechocystis SP PCC6803. XIV Convención y Feria Internacional Informática 2011.** La Habana, Cuba, del 7 al 11 de febrero de 2011.Publicado en: Programa Científico. Pág. 215**. ISBN 978-959-7213-01-7.**

Feist AM, Herrgard MJ, Thiele I, *et al.* (2009) **Reconstruction of biochemical networks in microorganisms.** Nat Rev Microbiology**, 7:**129-143**.**

Thiele I and Palsson BØ (2010). **A protocol for generating a high-quality genome-scale metabolic reconstruction.** Nat Protoc **5**, 93–121.

Karp PD, Paley S and Romero P, *et al* (2002). **The Pathway Tools software. Bioinformatics 2002,** 18 Suppl **1:**S225-32**.**

Duarte NC, Becker SA, Jamshidi N, *et al*. (2007) **Global reconstruction of the human metabolic network based on genomic and bibliomic data.** PNAS**, 104:**1777-1782**.**

Triana J, Córdova V, Jaime R, Reyes R, Garrido J, Villar L, Márquez F, Castro JC, Navarro E, Montagud A, F de Córdoba P, and Urchueguía JF *et al.* (2010). **Modelo metabólico de una cianobacteria, una fuente de energía a partir de la luz.** I Congreso Internacional de Ingeniería Química, Biotecnológica y Alimentaria (CIIQBA 2010).La Habana, Cuba, del 29 de noviembre al 3 de diciembre de 2010. Publicado en: Memorias de la Conferencia. **ISBN 978-959-261-317-1.**

Osterman A and Overbeek R, *et al* (2003). **Missing genes in metabolic pathways:a comparative genomics approach.** Curr Opin Chem Biol 2003**,7:**238-251**.**

Kharchenko P, Vitkup D and Church GM *et al*. (2004). **Filling gaps in a metabolicnetwork using expression information. Bioinformatics 2004,** 20Suppl **1:**I178-I185**.**

Kharchenko P, Chen L, Freund Y, Vitkup D and Church GM *et al*. (2006). **Identifyingmetabolic enzymes with multiple types of association evidence.BMC** Bioinformatics 2006**, 7:**177**.**

Chen L and Vitkup D, *et al* (2006). **Predicting genes for orphan metabolic activitiesusing phylogenetic profiles.** Genome Biol 2006**, 7:**R17**.**

Green ML and Karp PD, *et al* (2004). **A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases.** BMC Bioinformatics2004**, 5:**76**.**

Kumar VS, Dasika MS and Maranas CD et al. (2007). **Optimization based automated curation of metabolic reconstructions.** BMC Bioinformatics 2007, 8:212 doi**:10.1186/**1471-2105-8-212**.**

Reyes R, Gamermann D, Montagud A, Fuente D, Triana J, Urchueguía JF, Fernández de Córdoba P, *et al*. (2012) **Automation on the generation of genome scale metabolic models.** Journal of Computational Biology**, 19(12):** 1295-1306**.**

Triana J. *et* (2014) **MODEL-BASED ANALYSIS AND METABOLIC DESIGN OF A CYANOBACTERIUM FOR BIO-PRODUCTS SYNTHESIS. TESIS DOCTORAL.**

Reyes R *et* (2013). **DESARROLLO Y ANÁLISIS DE ALGORITMOS PROBABILÍSTICOS PARA LA RECONSTRUCCIÓN DE MODELOS METABÓLICOS A ESCALA GENÓMICA. TESIS DOCTORAL.**

Edwards JS, Ibarra RU and Palsson B *et al*. (2001). **In silico predictions of Escherichia coli metabolic capabilities are consistent with experimental data.** Nature Biotechnology**. 19:** 125-130**.**

Ibarra RU, Edwards JS and Palsson B *et al.* (2002). **Escherichia coli K-12 undergoes adaptive evolution to achieve in silico predicted optimal growth.** Nature 420 **| doi: 10.1038/nature01149.,**186-189**.**

Dandekar T, Moldenhauerc F, Bulikd S,Bertrama H and Schusterc S *et al*. (2003). **A method for classifying metabolites in topological pathway analyses based on minimization of pathway number.** BioSystems **70,** 255–270**.**

Hucka, M., Finney, A., Sauro, H.M., Bolouri, H., Doyle, J.C., Kitano, H., Arkin, A.P., Bornstein, B.J., Bray, D., Cornish-Bowden, A., Cuellar, A.A., Dronov, S., Gilles, E.D., Ginkel, M., Gor, V., Goryanin, I.I., Hedley, W.J., Hodgman, T.C., Hofmeyr, J.H., Hunter, P.J., Juty, N.S., Kasberger, J.L., Kremling, A., Kummer, U., Le Novre, N., Loew, L.M., Lucio, D., Mendes, P., Minch, E., Mjolsness, E.D., Nakayama, Y., Nelson, M.R., Nielsen, P.F., Sakurada, T., Scha,_ J.C., Shapiro, B.E., Shimizu, T.S., Spence, H.D., Stelling, J., Takahashi, K., Tomita, M., Wagner, J. and Wang, J. *et al* (2003). **The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models.** PubMed**, PMID:** 12611808.

Patil KR, Rocha I, Förster J, and Nielsen J. *et al* (2005) **Evolutionary programming as a platform for *in silico* metabolic engineering.** BMC Bioinformatics**, 6:**308**.**

Barrett C L, Kim T Y and Kim H *et al* (2006). **Systems biology as afoundation for genome-scale synthetic biology**. **Curr. Opin.** Biotechnol**.,17(5):**488–492**.**

Patil K R, Akesson M, Nielsen J *et al* (2004). **Use of genome-scale microbial modelsfor metabolic engineering. Curr.** Opin. Biotechnol**., 15(1):**64–69**,.**

Edwards J S, Ibarra R U and Palsson B O *et al* (2001). **In silico predictions of Escherichiacoli metabolic capabilities are consistent with experimental data.Nat.** Biotechnol**., 19(2):**125–130**.**

Oberhardt M A, Palsson B O and Papin J A *et al* (2009). **Applications of genomescalemetabolic reconstructions.** Mol. Syst. Biol**., 5:**320**.**

Montagud A, Navarro E, and Fernandez de Cordoba P *et al* (2010). **Reconstructionand analysis of genome-scale metabolic model of a photosynthetic bacterium.** BMC Syst Biol**, 4:**156**.**

Gamermann D, Montagud A and Conejero A *et al* (2014*)*. **Phylogenetic treereconstruction from genome-scale metabolic models. J.** Comput. Biol**.,21 (0):**1–12.

Varma A and Palsson B *et al (*1993*)*. **Metabolic capabilities of Escherichia coli: II.Optimal growth patterns. J.** Theor. Biol**., 165:**503–522**.**

Jeong H, Tombor B and Albert R *et al* (2000). **The large-scale organization ofmetabolic networks. Nature, 407(6804):651–654.**

Rives A W and Galitski T *et al* (2003). **Modular organization of cellular networks.** Proc. Natl. Acad. Sci. U.S.A**., 100(3):**1128–1133, Feb**.**

Ravasz E, Somera A L and Mongru D A *et al* (2002). **Hierarchical organizationof modularity in metabolic networks.** Science**, 297(5586):**1551–1555**.**

Schellenberger J, Que R, Fleming R M, Thiele I, Orth J D, Feist A M, Zielinski D C, Bordbar A, Lewis N E, Rahmanian S, Kang J, Hyduke D R and Palsson B O *et al* (2011). **Quantitative prediction of cellular metabolismwith constraint-based models: the COBRA Toolbox v2.0.** Nat Protoc**,6(9):**1290–1307**.**

Rocha I, Maia P and Evangelista P *et al* (2010). **OptFlux: an open-source softwareplatform for in silico metabolic engineering.** BMC Syst Biol**, 4:**45**.**

Hucka M, Finney A and Sauro H M *et al* (2003). **The systems biology markuplanguage (SBML): a medium for representation and exchange of biochemicalnetwork models.** Bioinformatics**, 19(4):**524–531**.**

Cvijovic M, Olivares-Hernandez R and Agren R *et al* (2010). **BioMet Toolbox:genome-wide analysis of metabolism.** Nucleic Acids Res**., 38(Web Serverissue):**W144–149**.**

Olivier B G, Rohwer J M and Hofmeyr J H S *et al* (2003). **Modelling cellular systemswith PySCeS.** Bioinformatics**, 21(4):**560–561**.**

Bastian M, Heymann S and Jacomy M *et al* (2013). **Gephi: An open source softwarefor exploring and manipulating networks.** International AAAI Conferenceon Weblogs and Social Media, 2009**.**

Bauer-Mehren A *et* (2013). **Integration of genomic information with biological networksusing Cytoscape.** Methods Mol. Biol**., 1021:**37–61**.**

Jünger M *et* (2004). **Graph drawing software.** Springer**, Berlin New York.**

Cock P J, Antao T, Chang T J, Chapman B A, Cox C J, Dalke A, Friedberg I, Hamelryck T, Kauff F, Wilczynski B *et al* (2009). **Biopython: freelyavailable python tools for computational molecular biology and bioinformatics.** Bioinformatics**, 25(11):**1422–1423**.**

Lopez C F, Muhlich J L, Bachman J A and Sorger P K *et al* (2013). **Programmingbiological models in python using pysb.** Molecular systems biology**, 9(1).**

Faircloth B C *et* (2008). **msatcommander: detection of microsatellite repeat arraysand automated, locus-specific primer design.** Molecular Ecology Resources**,8(1):**92–94**.**

Rybarczyk-Filho J L, Castro M A and Dalmolin R J *et al* (2011). **Towards agenome-wide transcriptogram: the Saccharomyces cerevisiae case.** NucleicAcids Res**., 39(8):**3005–3016**.**

Milne C B, Eddy J A and Raju R *et al* (2011). **Metabolic network reconstructionand genome-scale model of butanol-producing strain Clostridium beijerinckiiNCIMB 8052.** BMC Syst Biol**, 5:**130**.**

Klanchui A, Khannapho C and Phodee A *et al* (2012). **iAK692: a genome-scalemetabolic model of Spirulina platensis C1.** BMC Syst Biol**, 6:**71**.**

Amaral L A, Scala A, Barthelemy M and Stanley H E *et al* (2000). **Classes of smallworldnetworks.** Proc. Natl. Acad. Sci. U.S.A., **97(21):**11149–11152**.**

Paula MP *et al*. (2014). **Adaptación de herramientas de optimización monoobjetivo y multiobjetivo aplicadas a problemas de simulación de sistemas biológicos.** http://hdl.handle.net/10251/59531.

Mezura-Montes E, Reyes-Sierra M. and Coello CA. *et al* (2008). **Multi-objective Optimization Using Differential Evolution: A Survey of the State-of-the-Art.** Studies in Computational Intelligence book series **(SCI, volume 143)**

Reynoso-Meza G, Blasco X and Sanchis J, *et al* (2010). **Multiobjective optimization algorithm for solving constrained single objective problems**. Springer. Applications of Evolutionary Computation **pp 532-541**

Reynoso-Meza G, Blasco X, Sanchis J and Martínez M. *et al* (2014). **Controller tuning using evolutionary multi-objective optimisation: Current trends and applications.** Control Engineering Practice, **Volume 28, July 2014, Pages 58-73**

Navarro A and Boveris A. *et al* (2009). **Brain mitochondrial dysfunction and oxidative damage in Parkinson's disease**. Journal of Bioenergetics and Biomembranes. December 2009, **Volume 41, 6, pp 517–521**.

Tamagnini P, Leitão E, Oliveira P, Ferreira D, Pinto F, Harris DJ, Heidorn T and Lindblad P. *et al* (2007). **Cyanobacterial hydrogenases: diversity, regulation and applications.** Microbiology Reviews, Volume 31, 6, 1, **Pages 692–720**.

Tiwari JL and Terasaki PI, *et al* (2012). **HLA and Disease Associations**. Springer-Verlag. **New York Berlin Heidelberg Tokyo**

Petri CA *et* (1966). **Communication with Automata**." New York: Griffiss Air Force Base. Tech. Rep. **RADC-TR-65-377, vol. 1**.

Munkres JR *et* (2002) **Topología 2nd ed**., Pearson Educación, Madrid.

Baláž V, Kvasnička V and Pospíchal J. *et al* (1989). **Dual approach to edge distance between graphs**. Časopis pro pěstování matematiky, Vol. 114 (1989), **No. 2, 155--159**

Baláž V, Koča J, Kvasnička V and Sekanina M. *et al* (1986). **A metric for graphs**. Časopis pro pěstování matematiky, Vol. 111 (1986), No. **4, 431--433**

Wallis JD, Anderson KC and Miller EK, *et al* (2001). **Single neurons in prefrontal cortex encode abstract rules**. Nature, volume 411, **953–956**.

Vongsangnak W, Figueiredo LF, Förster J, Weber T, Thykaer J, Stegmann E,Wohlleben W and Nielsen J *et al* (2012). **Genome-scale metabolic representation of Amycolatopsis balhimycina**. Biotechnology and bioengineering. **109.** 1798-807. 10.1002/bit.24436.

Montagud A, Zelezniak A, Navarro E, Fernández de Córdoba P, Urchueguia J and Patil K *et al* (2011). **Flux coupling and transcriptional regulation within the metabolic network of the photosynthetic bacterium Synechocystis sp. PCC6803.** Biotechnology journal. **6**. 330-42. **10.1002/biot.201000109**.

Förster J, Famili I, Fu P, Ø Palsson B and Nielsen J *et al* (2003). **Genome-Scale Reconstruction of the Saccharomyces cerevisiae Metabolic Network.** Genome research. **13.** 244-53. 10.1101/gr.234503.

Borodina I, Krabben P and Nielsen J *et al* (2005), **Genome-scale analysis of Streptomyces coelicolor A3(2) metabolism**. Genome Res. 15, **820-829**

Triana J, Montagud A, Siurana M, Fuente D, Urchueguía A, Gamermann D, Torres J, Medialdea J, Fernández de Córdoba P and Urchueguia, J *et al* (2014). **Generation and Evaluation of a Genome-Scale Metabolic Network Model of Synechococcus elongatus PCC7942**. Metabolites. **4. 680-698. 10.3390/**metabo4030680.

Siurana M, Fernández de Córdoba P, Montagud A, Reynoso-Meza G, *et al* (2017). **Modelling and multi-objectiveoptimisation for simulation ofcyanobacterial metabolism. PhD** dissertation.

Bunke H and Shearer K *et al* (1998). **A graph distance metric based on the maximal common subgraph**. Pattern Recognition Letters, 19, **255–259.**

Bunke H (1997). **On a relation between graph edit distance and maximum common subgraph**. Pattern Recognition Letters, **18, 689-694**.

Apéndices

A comprehensive statistical study of metabolic and protein–protein interaction network properties

D. Gamermann [a], J. Triana-Dopico [b], R. Jaime [c]

[a] Department of Physics, Universidade Federal do Rio Grande do Sul (UFRGS) - Instituto de Física, Av. Bento Gonçalves 9500 - Caixa Postal 15051, CEP 91501-970, Porto Alegre, RS, Brazil
[b] Universidad Politécnica Salesiana - sede de Guayaquil, Ecuador
[c] Universidade Hermanos Saíz Montes de Oca - Pinar Del Rio, Cuba

HIGHLIGHTS

- Analysis done over thousands of graphs representing metabolic and PPI networks.
- The power-law (scale-free) function poorly describes the graphs degree distributions.
- The graph parameters are evaluated and compared to their expected values.
- Significant fluctuations are observed between the data and the expected values.
- Evolutionary models should drive other characteristics than the degree distributions.

ARTICLEINFO

Article history:
Received 13 December 2018
Received in revised form 24 April 2019
Available online 6 August 2019

Keywords:
Graphs
Biological networks
Degree distribution
PPI networks
Metabolic networks
Scale-free networks

ABSTRACT

Understanding the mathematical properties of graphs underlying biological systems could give hints on the evolutionary mechanisms behind these structures. In this article we perform a complete statistical analysis over thousands of graphs representing metabolic and protein–protein interaction (PPI) networks. First, we investigate the quality of fits obtained for the nodes degree distributions to power-law functions. This analysis suggests that a power-law distribution poorly describes the data except for the far right tail in the case of PPI networks. Next we obtain descriptive statistics for the main graph parameters and try to identify the properties that deviate from the expected values had the networks been built by randomly linking nodes with the same degree distribution. This survey identifies the properties of biological networks which are not solely the result of their degree distribution, but emerge from yet unidentified mechanisms other than those that drive these distributions. The findings suggest that, while PPI networks have properties that differ from their expected values in their randomized versions with great statistical significance, the differences for metabolic networks have a smaller statistical significance, though it is possible to identify some drift.

1. Introduction

Networks are an intuitive way to pictorially represent elements and their interactions in many complex systems. On top of its visual appeal, graph theory is a well established mathematical field which allows these structures to be quantitatively analyzed and have their properties objectively evaluated. As soon as the abstract graph theory began to be applied in order to describe real world networks, it became clear that graphs representing real systems differed vastly from what would be expected from the naive Ërdos–Rényi random model for networks studied in the fifties [1].

Graphs representing many different real world systems such as for example, author citations relations [2], biological networks [3,4] or flight connections [5] present many similar topological properties that significantly distinguish them from Ërdos–Rényi random graphs. Some of these common characteristics, often claimed to be ubiquitous in real world networks, are the presence of hubs (a few highly connected nodes), the so called small world property [6], scale-freeness or self-similarity as a consequence of the network's nodes degree distribution often be similar to a power-law function ($p(k) \propto k^{-\gamma}$) [7–9], high clusterization and hierarchical organization [10].

Biological systems are the result of evolution and natural selection. Therefore, the characteristics one observes in biological networks are indications of the evolutionary pressures under which these systems developed. The dense tails of the degree distributions in these networks, for example, have been suggested to give rise to the robustness of these systems against random node deletions [11–13], which would reflect the fact that live organisms are resilient to random mutations or to deprivation situations. In fact, many works study evolutionary models that attempt to generate graphs with similar degree distributions than the ones observed in real systems. These models define simple rules for the growth of a network and study the asymptotic behavior of an evolving graph. In the preferential attachment model [14–16], for example, a new node is more likely to connect to already highly connected nodes in the network, while in the duplication and divergence model [17] an existing node is first copied with the same connections and then may have its connections altered. These works usually focus in obtaining a set of rules that will generate random graphs with degree distributions similar to the ones observed in real systems. Therefore, it is important to determine the characteristics of these distributions on one hand and whether the degree distribution alone is enough to replicate other common characteristics of these networks such as high clusterization.

Though it is a very common claim that biological networks are scale-free (meaning that their node's degree distribution follows a power-law function), there are some studies that dispute this conclusion [18,19]. Many works that fit a powerlaw to the degree distribution of a given network, overlook the quality of the fit. However, in [20], an objective statistical study of several different real world networks is made in order to tackle this issue. In the study, the authors use robust statistical tools not only to fit the distributions but also to evaluate the quality of the fits (p-value), shedding light on which networks representing real systems might and might not be called scale-free, in the sense that their degree distributions follow the power-law function. In order to do that, the authors in [20] chose, for each real world system, a single representative network and analyze the fit obtained for its degree distribution. Though a single metabolic and a single protein–protein interaction (PPI) network are analyzed in the paper, these data sets composed of single elements are not enough to extract strong general conclusions about these biological systems. This study has been extended in [21] where the authors use the same fitting procedures in order to study bigger samples of networks and classify these different sets according to the evidence concerning the scale-free hypothesis. In this new study the authors study a much bigger sample of biological networks (675) than in the first study and conclude that most networks (~70%) are not scale-free or
have very week claim to be, but they do not present the results for the different classes of biological graphs metabolic, PPI, food web, anatomy, genetic regulation, . . .) separately. One may, therefore, not discard yet the hypothesis that a particular class of biological graphs is scale-free.

The authors of [22] note that to expect the degree distributions of these graphs to follow exactly a power-law ($P(k) \sim k^{-\gamma}$) might be a too restrictive hypothesis and propose to call a graph scale-free when its degree distribution is a regularly varying distribution ($P(k) = f(k)k^{-\gamma}$, where the function $f(k)$ varies slowly at infinity). This family of distributions would still have its right tail following a power-law function. The authors propose different estimators to evaluate the tail exponent, though they also remark that no hypothesis test might be devised in order to access the statistical significance of the results, in this case.

In [23] a version of the duplication and divergence model is analytically studied and it is observed that although it is possible to find parameter regions where the degree distribution will have a denser right tail, this distribution has a peak for low degrees indicating that it cannot be fitted to a monotonically decreasing power-law function if one considers its whole range.

The present study has performed two systematical analysis over two huge data sets of biological networks: more than 3000 metabolic and over 1000 PPI networks. The objective of the first analysis is to determine whether the degree distributions of the networks as a whole or their right tails may be well described as power-law (scale-free) functions. For this, we use the same tools as in [20] to fit the degree distribution of each network to a power-law and standard statistical tools in order to assess the quality of the obtained fits. In a second analysis, the main graph attributes of each network are evaluated, which allows us to draw a general picture about the characteristics of these graphs for a wide range of organisms. In order to identify which properties are simply expected given the degree distributions of the networks and which ones are the result of other possible evolutionary pressures over the underlying biological systems, the same properties are computed over randomized versions of each network that preserve their degree distributions. With this study, it is possible to evaluate the differences between the property in the real networks with respect to their randomized versions, to assess the statistical significance on the existence of such differences and, therefore, to identify attributes that are not only a consequence of the network's degree distribution.

The work is organized as follows: in the next section we define the graphs we study and from where the data in order to build them was retrieved. The section after that is dedicated to describe all the analysis done. Finally, we present the results and discussion and in the last section a brief overview and our conclusions. We also include an appendix on the distribution of a network characteristic in its randomized versions.

2. Data and graphs

In this section we define the graphs we analyze and describe the procedures followed in order to obtain the data from which we build the networks.

2.1. Metabolic networks

For an organism, we define its metabolic network as the undirected and unweighted graph resulting from connecting the molecules or metabolites appearing in its metabolism based on the biochemical reactions that keep its cells (or cell) alive. Two metabolites (nodes in the graph) are connected if they appear as a substrate-product pair in any chemical reaction in its metabolism.

Therefore, the data needed to build the metabolic network for one organism is the list of all biochemical reactions that can be found in its metabolism. This data was obtained from the Kyoto Encyclopedia of Genes and enomes (KEGG) database [24, 25]. First, a list of all genes in an organism is obtained, from this list of genes, those annotated as coding for enzymes are identified, as well as the corresponding chemical reactions catalyzed by each enzyme. In a complementary step, the pathways identified in each organism are obtained and the corresponding KGML files (KEGG Markup Language) are retrieved. These files allow one to identify the non-enzymatic reactions associated with known pathways. After the retrieval process, one has a list of chemical reactions from which one builds the network by listing all single metabolites appearing in the reactions and stipulating an undirected link between two metabolites whenever they appear in opposite sides of a reaction (as substrate-product).
An automated python script which connects to the database rest API was written, in order to obtain KEGG's list of organisms and run through it retrieving the data needed to build the networks [26]. The networks for 3481 organisms were successfully build by the procedure.

2.2. PPI networks

In a protein–protein interaction network, every protein found in an organism's proteome represents a graph node and two nodes are linked if the proteins have some kind of interaction between them.
Data for the production of PPI networks was downloaded from the STRING database [27]. From this database, for hundreds of organisms, one obtains lists of pairs of proteins present in the organisms and several scorings for each pair representing the confidence on the existence of some interaction between them (different scores are associated to different sources of evidence for the existence of the interaction). For all organisms downloaded, we built the network for each organism by setting the threshold on the minimal confidence level that an interaction must have in order to define an undirected link in the network. The threshold considered was 0.90 (in a range between 0 and 1) for the combined score. With this procedure we built 1073 PPI networks.

3. Statistical analysis

In this section we explain the graph parameters and characteristics that were analyzed in each network and the statistical tools employed in the analysis.
The theory on measurements related to graphs and the study of network characteristics and parameters can be found in several books and reviews. See, for example, [15,28].

3.1. Degree distribution

It is often claimed that biological networks have power-law (scale-free) node degree distribution. The discrete power-law distribution has the form:

$$\mathrm{p}(x/\gamma, x_0) = \begin{cases} \frac{x^{-\gamma}}{\zeta(\gamma, x_0)} & x \geq 0 \\ 0 & x < x_0 \end{cases} \quad (1)$$

$$\zeta(\gamma, x_0) = \sum_{x=x_0}^{\infty} x^{-\gamma} \quad (2)$$

where $\zeta(\gamma, x_0)$ is the Riemann zeta function (modified such that the sum starts at a minimum value x_0). This distribution has γ and x_0 as parameters. The parameter x_0 is an integer indicating the smallest number in the distribution.
According to the above, we attempt to fit a power-law (scale-free) distribution to the nodes degree distribution of the studied networks. In order to do that, given the set of N numbers $\{k_i, i = 1, 2, \ldots, N\}$ representing the

degree of every node in a network, we find the value of the parameter γ that maximize the likelihood, for a given value of x_0:

$$\ln \mathcal{L} = \sum_{i=1}^{N'} \ln p(k_i/\gamma, x_0) \tag{3}$$

where the sum is made over the degrees of every node in the network bigger than x_0. To find the parameter γ that maximizes the likelihood, one must solve the equation:

$$\frac{d}{d\gamma} \ln \mathcal{L} = 0 \tag{4}$$

$$N' \frac{\frac{d}{d\gamma}\zeta(\gamma, x_0)}{\zeta(\gamma, x_0)} - \sum_{i=1}^{N'} \ln k_i = 0 \tag{5}$$

where N' is the number of nodes in the network with degree bigger or equal to x_0 (which might be less than the total number of nodes, since nodes with degree less than x_0 are left out of the fit procedure).
Once γ is established, for many possible values x_0, we evaluate the goodness of fit through a x^2 test: the x^2 statistic is calculated and the right-cumulative distribution of the Pearson's x^2 distribution at this point is obtained. The result is the p-value i.e. the probability of obtaining a statistical fluctuation bigger than the observed one if the $k_i's$ distribution does come from a power-law with parameters γ and x_0. Therefore, big values of the p-value indicate a good fit.
For each network, we follow the same procedure: having its degree distribution, for every possible value of x_0 between 1 and some x_{max} we solve Eq. (5) and find the value of γ that maximize the likelihood for the given x_0. We also evaluate the p-value and count the amount (fraction) of nodes in the network with degree smaller than x_0 (these nodes did not participate in the fit procedure). We also evaluate an upper and lower uncertainty for the γ parameter by finding the two points around the maximum likelihood where it decreases half point (0.5) [29].
We also estimate the parameter γ, in each network, via the three estimators discussed in [22]: Hill [30], moments [31] and kernel [32]. These are estimators for the extreme value distribution index ξ, which relates to the exponent γ of the distribution via:

$$\xi = \frac{1}{\gamma - 1} \tag{6}$$

Note that a null or negative value for ξ indicates that the distribution is not regularly varying (scale-free) and, since there can be no hypothesis testing, as discussed in [22], the best one can hope for is to have the three estimators agreeing (have close, consistent values among them).

3.2. Graph properties

For every single graph produced (3481 metabolic networks and 1073 PPI networks), first the values for basic network parameters are obtained. Most of the obtained graphs contained small disconnected components, so we also count, for each graph, the number of disconnected components, the size of the biggest component and the average size of the smaller components. The most straight forward properties that are obtained from the graphs are its number of nodes N and its number of links $\mathcal{N}$.
Also, for every network it is straightforward to obtain its node's degree distribution i.e. n_i, the number of nodes in each network that have i links, for every possible integer i. This is evaluated by first obtaining the degree for each node i, k_i which is the number of links node i has:

$$k_i = \sum_{j=1}^{N} M_{ij} \tag{7}$$

where M_{ij} is the adjacency matrix of the graph (a square symmetrical $N \times N$ matrix where each element M_{ij} is 1 if node i is connected to node j and 0 otherwise).
Another local property of each node is its clustering coefficient C_i:

$$C_i = \frac{2\,E_i}{k_i(k_i - 1)} \tag{8}$$

where E_i is the number of links between the neighbors of node i. This coefficient is the ratio between the number of triangles node i actually forms with its neighbors and the total number of all possible ones given its degree k_i.
The local parameters (properties associated with each node in a network) can be averaged over all nodes in the network in order to establish an average network parameter. In the case of the two above mentioned parameters, one has the network's average degree $\bar{k}$ and average local clustering coefficient $\bar{C}$. It is also possible to define a global parameter representing the clustering of a network as the ratio between all triangles (size 3 clicks) the network (as a whole) actually has and the number of possible triangles it could have, based on the number of connected triples (length 2 paths):

$$c = 3\frac{|C_3|}{|P_2|} \tag{9}$$

where $|C_3|$ is the number of triangles (tree nodes connected in a cycle) and $|P_2|$ the number of 2-paths (connected triples).
We also study two parameters related to node correlations, namely nodes distances and network assortativity.
First we evaluate the symmetric distance matrix, a matrix where every element d_{ij} is shortest path length between node i to node j, via the Dijkstra's algorithm [33]. Since we consider the unweighted network (every link has weight 1), the size of the path is set as the distance between the two nodes. The average of all elements in the distance matrix is the network's average distance $\bar{d}$:

$$\bar{d} = \frac{2}{N(N-1)}\sum_{i=1}^{N}\sum_{j>i} d_{ij} \tag{10}$$

The network's assortativity, A, is a correlation coefficient between the node's excess degree and its expected value in an Ërdos–Rényi random network:

$$A = \frac{1}{\sigma_q^2}\sum_{k_i,k_j} k_i k_j (e(k_i,k_j) - q(k_i)q(k_j)) \tag{11}$$

$$q(k_i) = \frac{(k_i+1)p(k_i+1)}{\bar{k}} \tag{12}$$

where $p(k_i) = \frac{n_{k_i}}{N}$ is the probability that a node has degree k_i, $e(k_i,k_j)$ is the fraction of links in the network connecting nodes of degree k_i with k_j, $q(k_i)$ is called the excess degree distribution and σ_q is its standard deviation.
Positive coefficient A means an assortative network i.e. high degree nodes tend to be connected to other high degree nodes; while a negative coefficient A means a dissortative network that is, a network where high degree nodes tend to connect with low degree nodes.
We want to distinguish properties of the network that come solely as a natural consequence of its degree distribution and those that require some underlying mechanism to be achieved. To accomplish that, after having evaluated the network parameters, we compare those to averages of the same parameters evaluated over sets of randomized networks i.e. networks with the same degree distribution for their nodes, but where the links have been randomly exchanged. In the appendix we discuss the distribution of the network properties over the population of networks generated by this randomization process.
The randomization process we implemented is the following: first two links of the network are randomly selected. The links are broken and the nodes participating in one of the original links are connected to the nodes participating in the other original link. Since we work with simple undirected networks, sometimes this process fails, because given the randomly selected links, the relinking of the network would either generate a node connected with itself or two nodes sharing multiple connections. Repeating multiple times this process, one can also estimate (by bootstrap) the amount of times the process failed estimating, in this way, the probability of success in the process, which will be a property solely of the degree distribution of the network. We call this parameter η. Note that this randomization process does not change the degree distribution of the network i.e. every node keeps its k_i constant during the process.
One is left to decide how many times to repeat this randomization process in order to obtain a truly random network. We adopt the paradigm that each link should have a 99.9% probability of having been touched at least once by the process. The idea behind this process is to generate random networks with the same degree distribution as the original network therefore, assessing the properties of these random networks, one obtains the characteristics of the network that emerge only as a consequence of the degree distribution of the graphs. This set of random networks mimics what would be expected as the result from an evolutionary model constructed in order to generate networks adjusting the degree distribution observed in real world graphs and not caring with any other aspect of the resulting graphs.

Table 1
Fitted parameter and uncertainties averaged using p-value as weight, for metabolic networks. In parenthesis, next to the average p-value, the fraction of fits for which the individual p-value was greater than 0.01. In the column Discarded the fraction of nodes whose degree is less than x_0, and therefore did not participated in the calculations, is presented.

x_0	γ	Discarded	Average p-value (fraction > 0.01)
1	$1.569968^{+0.057497}_{-0.053732}$	0.000000	0.000000 ± 0.000000 (0.000000%)
2	$2.134354^{+0.061622}_{-0.059194}$	0.045531	0.012007 ± 0.037116 (58.527493%)
3	$2.372021^{+0.139983}_{-0.130618}$	0.358368	0.003266 ± 0.026371 (12.395154%)
4	$2.654597^{+0.214409}_{-0.196739}$	0.519733	0.001849 ± 0.021633 (6.337372%)
5	$2.592453^{+0.251536}_{-0.226789}$	0.693625	0.001751 ± 0.018163 (7.362535%)
6	$2.730658^{+0.324567}_{-0.287168}$	0.761719	0.001357 ± 0.015890 (6.150979%)
7	$2.700261^{+0.356084}_{-0.311195}$	0.820451	0.001273 ± 0.013544 (7.362535%)
8	$2.797779^{+0.407363}_{-0.352410}$	0.849825	0.000934 ± 0.010066 (5.498602%)
9	$2.794249^{+0.449053}_{-0.383302}$	0.880596	0.000782 ± 0.007931 (4.473439%)
10	$2.863253^{+0.504359}_{-0.425526}$	0.898041	0.000683 ± 0.006580 (4.287046%)
11	$2.910327^{+0.552005}_{-0.460801}$	0.911973	0.000647 ± 0.005911 (4.380242%)
12	$3.006127^{+0.618104}_{-0.509971}$	0.921416	0.000545 ± 0.005157 (3.914259%)
13	$3.102794^{+0.690538}_{-0.562881}$	0.929619	0.000455 ± 0.004476 (3.075489%)
14	$3.166999^{+0.753881}_{-0.607058}$	0.936420	0.000414 ± 0.004488 (2.423113%)
15	$3.429787^{+0.859267}_{-0.689126}$	0.937253	0.000431 ± 0.005674 (2.329916%)

Since in each step of the randomization process two links are selected, the probability p that any given link is touched in a given step is $p = \frac{2}{\mathcal{N}}$. Therefore, the probability that a link is not touched is $\bar{p} = 1 - p$. If the randomization process is repeated n times, the probability that a given link is never touched is p^{-n}. So the number of times we must repeat the randomization process in order that there is a 99.9% probability that any given link was touched by the process (probability $\frac{1}{1000}$ of not being touched) is:

$$n = -\frac{\ln(1000)}{\ln\left(1 - \frac{2}{\mathcal{N}}\right)} \tag{13}$$

to give an idea of his number, in a typical metabolic network with 2400 links, this value is $n = 8286$, while for a typical PPI network with 28 000 links, this number is $n = 96\,705$.

For each network, we obtain 10 randomized versions of it, evaluate each network parameter in each randomized network and estimate an expected value and its uncertainty by evaluating the average and standard deviation of the parameter over the ten randomized network samples. In this way, we are able to evaluate, for each network, a measure of the parameter deviation in the real network from the expected value in the randomized versions of it, by computing the statistical t_p for each parameter P:

$$t_p = \frac{\bar{p}_{ramdon} - \bar{p}_{real}}{\frac{s}{\sqrt{10}}} \tag{14}$$

where $\bar{p}_{real}$ is the value obtained for the parameter P in the real network, $\bar{p}_{ramdon}$ is the average value of the parameter over the ten randomized versions of the network and S the standard deviation of the parameter in the random samples.

The statistical t_p can be used to assess the statistical significance on the existence of a difference between the parameter value in the real network and its expected value in randomized versions of it, by evaluating the cumulative student's t-distribution with 9 degrees of freedom at point $|t_p|$. In the appendix we provide a normality test for the distribution of the network characteristics over its randomized versions, justifying thus the use of the student's t-test. Two times the value of this cumulative distribution is interpreted as the p-value for the null hypothesis that the observed network has the P parameter equal to its expected value. High values of this p-value would indicate that the difference is not significant (high probability of obtaining a fluctuation equal or bigger than the observed one in the population). Parameters for which the p-value is small would indicate a significant difference between the observed and expected value, hinting that evolution favors (selects) networks in which the parameter value is bigger (if t_p is negative) or smaller (if t_p is positive) than what is expected in a random version of the network. Therefore, a good dynamical evolutionary model for these biological systems would have to incorporate mechanisms that result in networks with such characteristics.

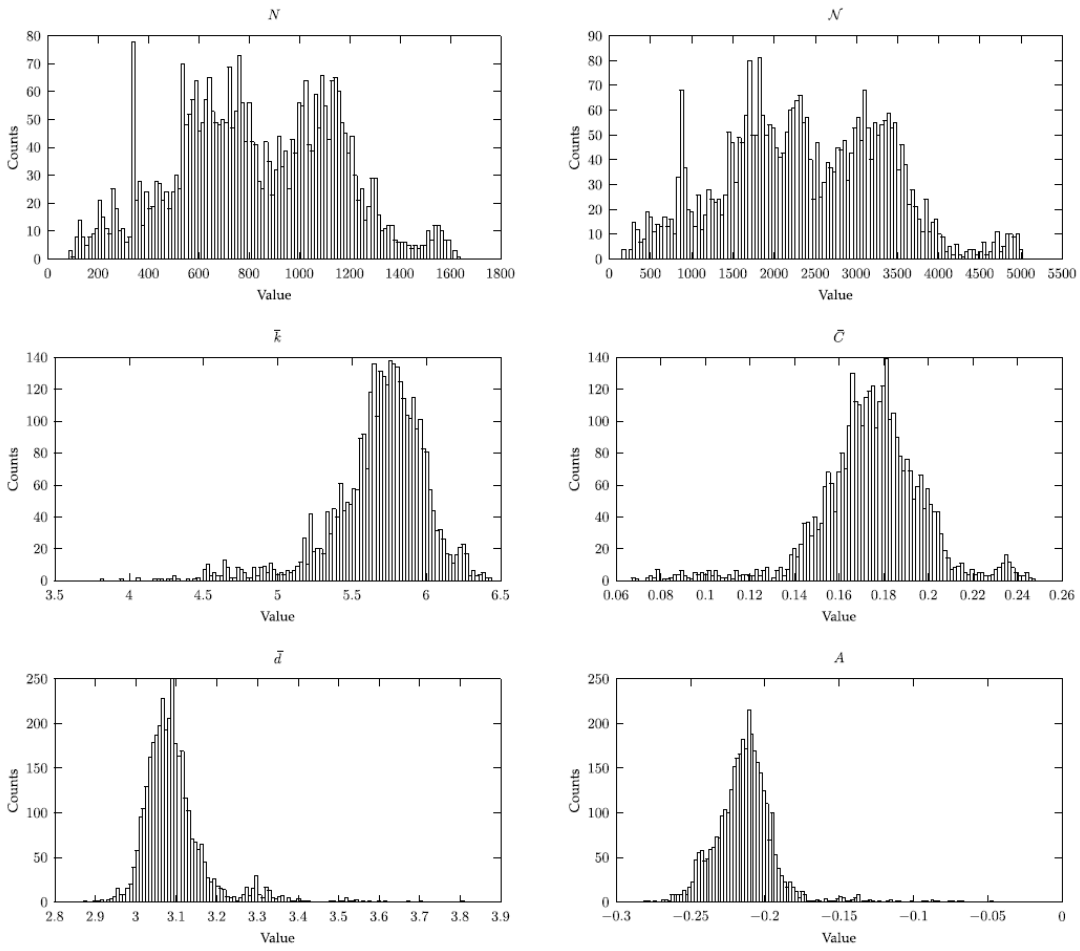


Fig. 1. Histograms for the parameter distributions in metabolic networks. Top left, right: number of nodes and number of links. Center left, right: average degree and average local clustering. Bottom left, right: Average shortest path and assortativity.

4. Results and discussion

For each data set of networks studied (metabolic and PPI), we present first the results for fitting procedure for the node's degree distributions followed by the descriptive statistics for the network parameters and then the results from the comparison between the real and the randomized network versions.

4.1. Metabolic networks

In Table 1 we show the results for the fitting procedure done over the node's degree distributions, averaged over the 3481 metabolic networks. For each network, for each value of x_0 between 1 and 15, we evaluate the value for the parameter γ that maximizes the likelihood for the observed degrees in the network (solution of Eq. (5)), we obtain its uncertainties, and for the fitted value of γ the statistical x_2 is computed along with its correspondent p-value. Note that the fit procedure is done for every x_0 in the table for each one of the 3481 graphs in the data set. Therefore, the values of γ and its uncertainties presented in the table are the result of averaging the obtained values of γ and its uncertainty in each network weightening the values by the p-value of the fits (giving more importance to the better fitted values). Next to the average p-value we also show the fraction of the fits for which the p-value was bigger than 1%. Note also that, for each value of x_0 bigger than 1, some nodes (the ones for which $k_i < x_0$) are left out of the fit. The table presents also the average fraction of discarded nodes in the fits, for each value of x_0.

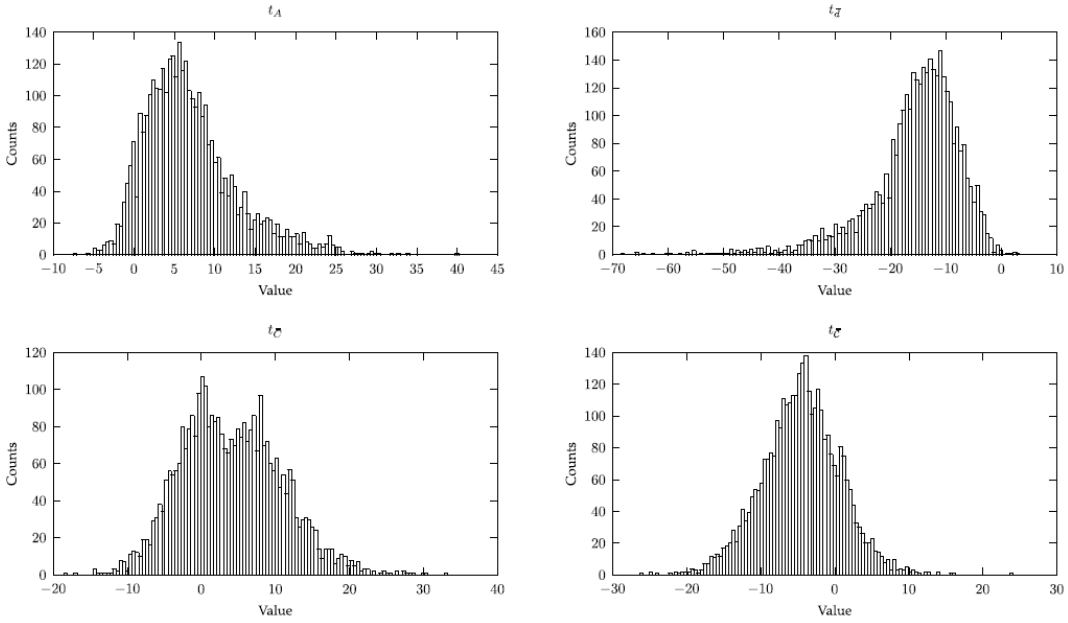


Fig. 2. Histograms for the distribution of the parameter t_p in metabolic networks. Top left, right: assortativity and average shortest path. Bottom left, right: Average local clustering and global clustering.

Table 2
Average values for the γ obtained via the three estimators for the extreme value distribution index. The first two columns show the estimator name and its symbol as used in the present work. Around the average, shown as uncertainties, are the standard deviations calculated for values bigger and smaller than the average, separately. The last column is the proportion of networks for which a $\xi < 0$ was obtained.

Estimator	Math symbol	Average	Standard deviation	Skewness	Discarded
Hill	γ_{hill}	$2.529634^{+0.144538}_{-0.061708}$	0.105341	12.753512	0.000000
Moments	$\gamma_{moments}$	$2.379529^{+0.544833}_{-0.126478}$	0.359907	18.646036	0.019822
Kernel	γ_{kernel}	$2.453754^{+1.202333}_{-0.256618}$	0.689289	12.067175	0.000287

each network weightening the values by the p-value of the fits (giving more importance to the better fitted values). Next to the average p-value we also show the fraction of the fits for which the p-value was bigger than 1%. Note also that, for each value of x_0 bigger than 1, some nodes (the ones for which $k_i > x_0$) are left out of the fit. The table presents also the average fraction of discarded nodes in the fits, for each value of x_0.

For $x_0 = 1$, the p-value is lower than 10^{-6}, indicating that only the tail of the distribution might be well adjusted to a power-law function and the highest p-values obtained are for $x_0 = 2$. In this case, the value of γ is a little above 2.1 and around 4% of the nodes in the network have degree 1 and do not participate in the fit. But even in this case, the average p-value is only around 0.01 indicating that the deviation from a power-law is still significant.

Next, applying the tools discussed in [22], we obtained for each metabolic network, the three γ values correspondent to the three ξ estimators. These are in Table 2. In a few cases, the ξ estimator obtained was negative, which indicates that the distribution is not well described as scale-free. From the table, one sees that the three averages obtained are consistent among them and in very few cases the ξ value was negative. This indicates that the distributions can indeed be classified as scale-free. The values of γ obtained in these cases are consistent with the previous analysis and values found for $x_0 = 2$ and $x_0 = 3$.

Now, for each one of the 3481 metabolic networks in our data set, we evaluated the graph properties and characteristics described in Section 3.2. Histograms depicting the distributions of the main parameters over our data set are shown in Fig. 1. The distributions show the bulk of the data distributed around a central value, but all of them also present a significant number of outliers.

In Table 3 we present the descriptive statistic for the parameters. Since some distributions have a sizable skewness (asymmetry), besides evaluating the standard deviation of the distribution, we also evaluated the standard deviation for all values bigger and smaller than the average, separately. These are shown in the table as uncertainties around the average value of each parameter.

Table 3
Descriptive statistics for the distribution of metabolic network parameters. The first two columns show the parameter name (definition) and its symbol as used in the present work. Around the average, shown as uncertainties, are the standard deviations calculated for values bigger and smaller than the average, separately.

Parameter	Math symbol	Average	Standard deviation	Skewness
Number of nodes	N	$827.228670^{+327.515437}_{-313.031642}$	319.999188	0.000736
Number of links	$\mathcal{N}$	$2384.474864^{+1017.416129}_{-951.857140}$	983.598923	0.093452
Average degree	$\bar{k}$	$5.686846^{+0.248550}_{-0.400098}$	0.321772	−1.359436
Average local clustering	$\bar{C}$	$0.174469^{+0.021995}_{-0.026009}$	0.024008	−0.658240
Global clustering	C	$0.052559^{+0.014486}_{-0.011211}$	0.012855	0.776866
Assortativity	A	$-0.213441^{+0.022995}_{-0.018595}$	0.020923	1.587561
Average distance	$\bar{d}$	$3.094789^{+0.110611}_{-0.054124}$	0.081075	2.340936
Average number of 2-paths	P_2	$104\,553.170928^{+91543.797126}_{-59024.598610}$	74647.830744	1.090137
Average number of triangles	C_3	$1564.592933^{+916.729975}_{-748.085279}$	830.036302	0.520495
Average number of components	N_{comps}	$7.202528^{+5.224926}_{-3.526826}$	4.316908	0.762602
Average size of main component	$\mathcal{S}_{main}$	$813.934502^{+318.963107}_{-306.925997}$	312.720429	−0.008834
Average size smaller components	$\mathcal{S}_{small}$	$2.053929^{+0.487501}_{-0.470099}$	0.475511	−2.004091

Table 4
Parameter values in randomized versions of the metabolic networks.

Parameter	Math symbol	Average	Standard deviation	Skewness
Average local clustering	$\bar{C}_{rand}$	$0.184840^{+0.021231}_{-0.025024}$	0.023123	−0.548522
Global clustering	C_{rand}	$0.049392^{+0.014380}_{-0.009870}$	0.012134	1.112934
Assortativity	A_{rand}	$-0.206991^{+0.020612}_{-0.015880}$	0.018317	1.247442
Average distance	$\bar{d}_{rand}$	$3.013782^{+0.057361}_{-0.050978}$	0.054262	0.760095
Average number of triangles	C_{3rand}	$1469.290822^{+875.345398}_{-705.010822}$	787.625049	0.521137
Average number of components	$N_{compsrand}$	$1.255544^{+0.263182}_{-0.162395}$	0.208086	0.892551
Average size main Component	$\mathcal{S}_{mainrand}$	$826.693292^{+327.009958}_{-312.930479}$	319.708235	0.000727
Average size of smaller components	$\mathcal{S}_{smallrand}$	$0.450593^{+0.380467}_{-0.285910}$	0.331312	0.586352
Probability of success in the randomization process	η	$0.866479^{+0.012417}_{-0.011787}$	0.012108	0.307997

Table 5
Deviation of real metabolic network parameters from the randomized expected values. The column p-value presents the average p-value for the student's t-test evaluated as described in Section 3.3. In the parenthesis next to it, it is presented the fraction of networks for which the p-value was below 0.05.

t	Average	Standard deviation	Skewness	p-val (fraction < 0.05)
$t_{\bar{C}}$	$3.998663^{+7.164259}_{-6.293898}$	6.723294	0.319300	0.106481 (74.289%)
$t_{\bar{C}}$	$-4.565913^{+5.219167}_{-5.353847}$	5.285228	−0.013612	0.086937 (76.300%)
$t_{\bar{d}}$	$-15.517486^{+6.222337}_{-10.792203}$	8.417728	−1.588110	0.002822 (99.138%)
t_A	$6.882841^{+6.991963}_{-4.492181}$	5.685528	1.056430	0.072785 (81.270%)
t_{C_3}	$-4.565913^{+5.219167}_{-5.353847}$	5.285228	−0.013612	0.086937 (76.300%)

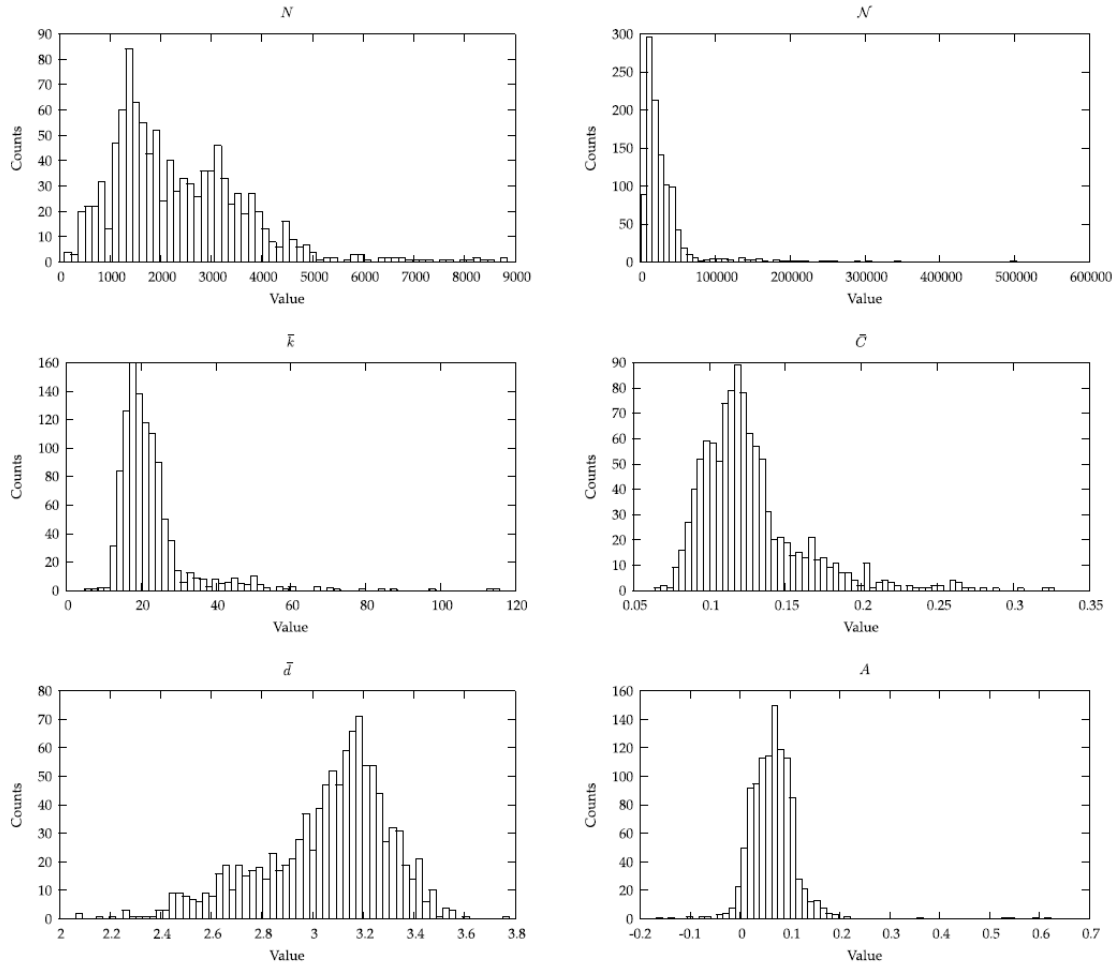


Fig. 3. Histograms for the parameters in PPI networks. Top left, right: number of nodes and number of links. Center left, right: average degree and average clustering. Bottom left, right: Average shortest path and assortativity.

In a real metabolic network, one would not expect disconnected components. The small components into which the networks fragment themselves are possibly a problem of wrong annotations in the databases or misidentification of some chemical reactions or metabolites within them in the automated process of reconstructing the networks. In any case, as can be seen from the difference between the average size of the main components and the average total number of nodes, the disconnected components amount to a negligible number of nodes.
All metabolic networks are dissortative $(A < 0)$. It is also possible to observe that, while the networks show a high average local clustering, their global clustering parameter tend to be small, close to zero. The networks tend to cluster locally, but not globally. The lack of correlation between the local and global clustering coefficients in real networks has already been observed in other systems [28,34].
Table 4 shows the same parameters as in Table 3 (except for those that are not affected by the randomization process, like $N, \mathcal{N}$ or p_2) evaluated for the average values of the randomized samples of each network. The last row in this table presents the parameter η, whose average value is around 0.87, indicating that, on average, in 13% of the randomization steps, the links broken could not have been properly relinked (the random step failed).
The statistical significance of the differences between the parameters in Tables 3 and 4 can be appreciated in Table 5 where it is shown the average value for the statistical t_p for each parameter. For each network (3481 in total) the value of t_p and its correspondent p-value is evaluated. The table shows the statics of the distribution of t_p over the whole data set. Next to the average p-value, we also present the fraction of the networks for which this p-value was below 0.05.
Histograms for the distribution of t_p for the different parameters can be found in Fig. 2.

Table 6
Fitted parameter and uncertainties weight averaged by p-value for PPI networks. In parenthesis, next to the average p-value, the fraction of fits for which the individual p-value was greater than 0.01. In the column Discarded the fraction of nodes whose degree is less than x_0, and therefore did not participated in the calculations, is presented.

x_0	γ	Discarded	Average p-value (fraction > 0.01)
2	$1.865963^{+0.099169}_{-0.091994}$	0.192307	0.000000 ± 0.000000 (0.000000%)
3	$2.221444^{+0.151157}_{-0.139448}$	0.307692	0.000000 ± 0.000001 (0.000000%)
4	$2.439663^{+0.204404}_{-0.186506}$	0.470854	0.000000 ± 0.000003 (0.000000%)
5	$2.832902^{+0.281938}_{-0.255359}$	0.545912	0.000002 ± 0.000077 (0.000000%)
6	$3.242113^{+0.375284}_{-0.336876}$	0.610980	0.000013 ± 0.000402 (0.093197%)
7	$3.450916^{+0.474926}_{-0.419303}$	0.701252	0.000015 ± 0.000455 (0.093197%)
8	$3.898085^{+0.560377}_{-0.491327}$	0.682979	0.000110 ± 0.002786 (0.186393%)
9	$3.130937^{+0.321283}_{-0.280358}$	0.586882	0.000206 ± 0.003768 (0.279590%)
10	$2.455655^{+0.104328}_{-0.094032}$	0.534252	0.000896 ± 0.012494 (1.025163%)
11	$2.439406^{+0.102351}_{-0.089275}$	0.559666	0.002384 ± 0.023471 (1.863933%)
12	$2.476708^{+0.090556}_{-0.080428}$	0.578328	0.005589 ± 0.045938 (3.727866%)
13	$2.475588^{+0.070015}_{-0.064430}$	0.591279	0.011535 ± 0.073171 (6.150979%)
14	$2.505834^{+0.058103}_{-0.056443}$	0.599493	0.021006 ± 0.100260 (11.649581%)
15	$2.535557^{+0.059400}_{-0.057562}$	0.601718	0.037984 ± 0.132088 (18.732526%)
16	$2.563771^{+0.058223}_{-0.056523}$	0.603065	0.066005 ± 0.177740 (27.772600%)
17	$2.607947^{+0.059156}_{-0.056975}$	0.612704	0.102540 ± 0.226509 (32.898416%)
18	$2.665189^{+0.062425}_{-0.060167}$	0.628512	0.141075 ± 0.269827 (38.956198%)
19	$2.713628^{+0.066267}_{-0.064029}$	0.644746	0.177258 ± 0.303147 (45.013979%)
20	$2.765948^{+0.069993}_{-0.066823}$	0.660004	0.212718 ± 0.330160 (49.953402%)
21	$2.820009^{+0.074240}_{-0.071932}$	0.673400	0.248691 ± 0.351983 (54.426841%)
22	$2.870172^{+0.076718}_{-0.074182}$	0.686005	0.282034 ± 0.368536 (58.434296%)
23	$2.921708^{+0.080703}_{-0.078049}$	0.697715	0.309604 ± 0.379620 (62.068966%)
24	$2.971699^{+0.085828}_{-0.082439}$	0.708591	0.338036 ± 0.388383 (64.958062%)
25	$3.017648^{+0.088027}_{-0.085234}$	0.718502	0.364349 ± 0.395045 (67.567568%)
26	$3.063582^{+0.091597}_{-0.088628}$	0.727871	0.384254 ± 0.398943 (69.897484%)
27	$3.111638^{+0.095501}_{-0.092096}$	0.736455	0.401460 ± 0.401592 (72.320596%)
28	$3.153570^{+0.098928}_{-0.095350}$	0.743871	0.412428 ± 0.401466 (74.650513%)
29	$3.193774^{+0.102076}_{-0.097709}$	0.750596	0.419412 ± 0.401385 (75.955266%)
30	$3.230413^{+0.105087}_{-0.099465}$	0.757299	0.424672 ± 0.401179 (76.514445%)
31	$3.268202^{+0.108429}_{-0.103152}$	0.764223	0.426257 ± 0.399990 (77.446412%)
32	$3.302613^{+0.111565}_{-0.104587}$	0.770964	0.424518 ± 0.398700 (78.285182%)
33	$3.338181^{+0.115467}_{-0.108414}$	0.777333	0.419403 ± 0.398042 (78.378378%)
34	$3.370140^{+0.118337}_{-0.110493}$	0.783326	0.413641 ± 0.398115 (78.657968%)
35	$3.399909^{+0.120654}_{-0.113122}$	0.788990	0.405643 ± 0.398281 (78.191985%)
36	$3.429541^{+0.123410}_{-0.115413}$	0.793904	0.395080 ± 0.397048 (78.098788%)

(*continued on next page*)

The most significant difference observed is for the average shortest path. Note, that the significance in this analysis is not on the amount of the difference, but on the confidence level for the existence of this difference. If one compares the difference between $\bar{d}$ and $\bar{d}_{rand}$ in Tables 3 and 4 it amounts to around 2.5%. Though this difference is small, given any metabolic network, one can say with high confidence that the average shortest path is bigger in the real network than its expected value in randomized versions of it. For the other parameters the situation is not as clear. One observes that a sizable fraction of the networks (around $3/4$ of them) do show significance on the existence of some difference ($p - val < 0.05$), but these differences are not ubiquitous.

Table 6 (continued).

x_0	γ	Discarded	Average p-value (fraction > 0.01)
37	$3.457949^{+0.126062}_{-0.117349}$	0.798374	0.383044 ± 0.394640 (77.912395%)
38	$3.483018^{+0.128546}_{-0.117759}$	0.802189	0.372147 ± 0.393137 (76.421249%)
39	$3.507460^{+0.131886}_{-0.120585}$	0.805173	0.358892 ± 0.390367 (75.582479%)
40	$3.533017^{+0.134452}_{-0.123458}$	0.808156	0.347501 ± 0.387838 (73.904939%)

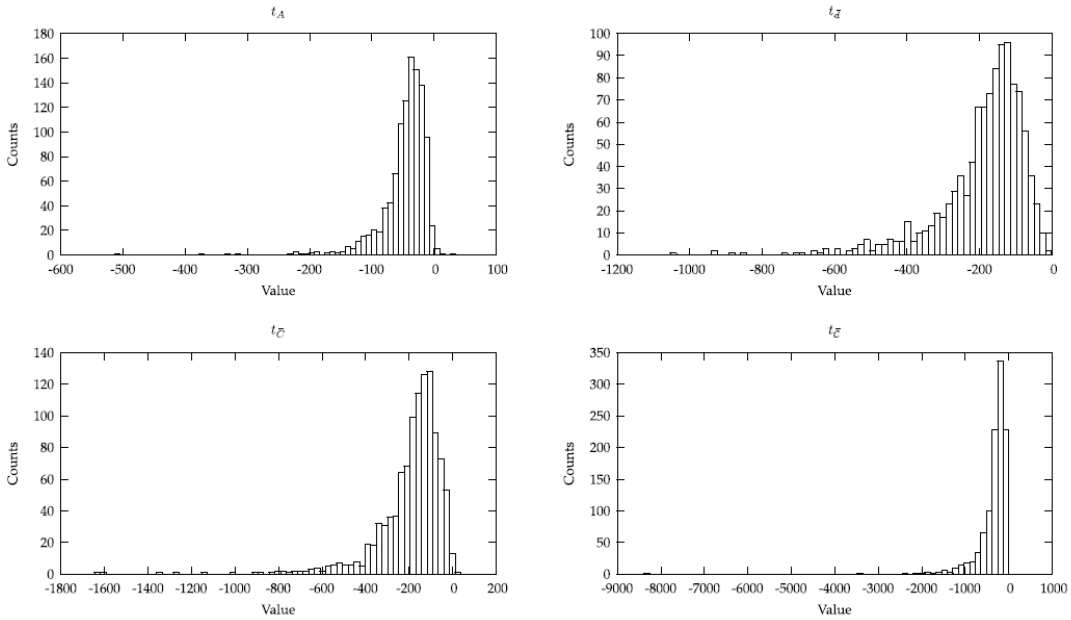


Fig. 4. Histograms for parameter deviations in PPI networks. Top left, right: assortativity and average shortest path. Bottom left, right: Average local clustering and global clustering.

4.2. PPI networks

The same analysis were done over the data set of 1073 protein–protein interaction (PPI) networks. First, in Table 6 we present the results of the fit procedure for the degree distributions. Here, one can see that it is not possible to obtain a reasonable fit unless more than 50% of the network's nodes are left out of the fit. Only for the far right tail of the distribution (around $x_0 \geq 13$) one begins to obtain reasonable quality fits. In particular, for $x_0 = 1$ the fits were so bad, that the p-values were all smaller than the machine precision ($p < 10^{-16}$) and therefore it was not possible to evaluate the averages weighted by the p-values.

In Table 7, we present the results for the γ values obtained via the three estimators of the distributions' extreme value indexes. Note that now, the average values are discrepant with one another. Moreover, for many networks (around 50% in the case of the Kernel estimator), the value of ξ was lower than zero. The distributions of these estimators in the case of PPI networks present huge differences from network to network and therefore, no consistence among the different graphs. Based on the discussions in [22], one would conclude that the distributions are not regularly varying (scale-free) ones. The previous analysis, on the other hand, shows that the right tail (around 25%) of the degree distributions of PPI networks are well fitted by a power-law function.

Next we performed the analysis of the network properties of the data set. In Fig. 3 we present the histograms for the distributions of the main graph parameters and in Table 8, the computed descriptive statistics of the distributions.

The PPI networks show some peculiar properties if compared with the metabolic ones. They present both, local and global clustering coefficients significantly different from zero; These networks are mildly assortative and, though they are bigger in size, they are more densely connected and present a similar shortest average path to the metabolic graphs.

Table 7

Average values for the γ obtained via the three estimators for the extreme value distribution index. The first two columns show the estimator name and its symbol as used in the present work. Around the average, shown as uncertainties, are the standard deviations calculated for values bigger and smaller than the average, separately. The last column is the proportion of networks for which a $\xi < 0$ was obtained.

Estimator	Math symbol	Average	Standard deviation	Skewness	Discarded
Hill	γ_{hill}	$6.699140^{+43.143903}_{-2.123480}$	15.826346	20.817997	0.000000
Moments	$\gamma_{moments}$	$17.697194^{+100.650635}_{-9.515008}$	43.782798	9.634417	0.184529
Kernel	γ_{kernel}	$72.375354^{+2101.436627}_{-59.110375}$	623.100839	18.278472	0.496738

Table 8

Descriptive statistics for the distribution of PPI network parameters.

Parameter	Math symbol	Average	Standard deviation	Skewness
Number of nodes	N	$2378.162162^{+1674.170267}_{-1062.364609}$	1360.967079	1.312170
Number of links	$\mathcal{N}$	$30\,648.163094^{+63095.796878}_{-16283.411100}$	38 129.530762	5.244932
Average degree	$\bar{k}$	$22.660161^{+16.152767}_{-5.400247}$	10.321305	3.606450
Average local clustering	$\bar{C}$	$0.126916^{+0.048151}_{-0.023306}$	0.034739	1.852084
Global clustering	$\mathcal{C}$	$0.103259^{+0.077102}_{-0.023378}$	0.046880	4.257505
Assortativity	A	$0.065945^{+0.060521}_{-0.042112}$	0.051916	3.538212
Average distance	$\bar{d}$	$3.053442^{+0.200444}_{-0.315897}$	0.253305	−0.846669
Average number of 2-paths	P_2	$1\,998\,518.199441^{+12364896.830628}_{-1324465.387398}$	5 674 983.876744	11.761741
Average number of triangles	C_3	$65\,643.418453^{+378962.677641}_{-44543.188780}$	176 952.116048	11.214598
Average number of components	N_{comps}	$3.057782^{+2.966847}_{-1.430859}$	2.064812	1.279759
Average size of main component	$\mathcal{S}_{main}$	$2372.873253^{+1672.813536}_{-1060.363478}$	1359.360376	1.317361
Average size of smaller components	$\mathcal{S}_{small}$	$1.915665^{+1.255322}_{-1.919120}$	1.455628	1.267565

Table 9 shows the descriptive statistics for the average values of the parameters obtained from the randomized samples.

These values show some sizable differences with respect to the parameters in the real networks shown in Table 8. For the random samples, the global and local clustering coefficients are close to zero. The different behavior of the global clustering coefficient can be directly linked to the number of triangles in the networks: the real networks present almost twice more size 3 clicks than the random samples. Moreover, the networks present a mild dissortative degree correlation and a slightly smaller average shortest path. The statistical significance of the differences can be read in Table 10, where we present the descriptive statics for the $t_p$statisticals and their associated averaged p-values. In Fig. 4 are depicted the histograms for the distributions of these statisticals.

In the case of PPI networks, the differences between real and expected values for all parameters are statistically significant. Real PPI networks present bigger average local and global clustering coefficients than their expected values in random networks with the same degree distribution, as well as bigger shortest average paths and assortativity. Realistic evolutionary models that try to mimic growth of PPI networks should not only try to reproduce their degree distribution (which is not well described by a power-law function) but also try to incorporate underlying mechanisms that result in networks with such deviations from the expected values in networks with the same degree distributions.

5. Overview and conclusion

We have analyzed a huge set of graphs representing two classes of biological systems (3481 metabolic networks and 1073 PPI networks). First, we study in detail the degree distributions of the networks and test them against the hypothesis that they follow a power-law function. The results of this first analysis show that the degree distributions of these real world graphs are not well described by this function, but only the right tail containing a small fraction of the nodes in the case of PPI networks are reasonably adjusted to this scale-free distribution. The scale-free hypothesis was also tackled from the less restrictive hypothesis that the degree distributions are regularly varying distributions. This analysis resulted in different estimators for the exponent γ of the tail of the distributions consistent among themselves in the case of metabolic networks, but not in the case of PPI graphs. In a second analysis, a complete descriptive statistics of the networks properties is presented and then we identify those parameters that deviate from their expected values in randomized versions of the graphs that preserve the network's degree distribution. Biological networks are the result of evolution and natural selection. Therefore, these deviations are the result of evolutionary pressures these systems developed under. Realistic evolutionary models that describe these systems should incorporate mechanisms that result in graphs with such deviations and not only try to reproduce their degree distributions.

Table 9
Parameter values in randomized versions of the PPI networks.

Parameter	Math symbol	Average	Standard deviation	Skewness
Average local clustering	$\bar{C}_{rand}$	$0.043989^{+0.041802}_{-0.012590}$	0.025241	3.782684
Global clustering	C_{rand}	$0.043776^{+0.038540}_{-0.012340}$	0.023717	3.300061
Assortativity	A_{rand}	$-0.024558^{+0.018546}_{-0.013473}$	0.016204	2.909341
Average distance	$\bar{d}_{rand}$	$2.765340^{+0.201126}_{-0.272347}$	0.235211	−0.607108
Average number of triangles	C_{3rand}	$28\,250.808760^{+285093.382085}_{-18602.848604}$	106 267.234720	14.761007
Average number of components	$N_{compsrand}$	$1.352190^{+0.622019}_{-0.255707}$	0.425907	2.844741
Average size of main component	$\mathcal{S}_{mainrand}$	$2377.441473^{+1673.836734}_{-1062.029476}$	1360.641365	1.312565
Average size of smaller components	$\mathcal{S}_{smallrand}$	$0.529844^{+0.628106}_{-0.387873}$	0.507262	1.765283
Probability of success in the randomization process	η	$0.982054^{+0.007652}_{-0.038223}$	0.020064	−6.207947

Table 10
Deviation of real PPI network parameters from randomized expected values. The column p-value presents the average p-value for the student's t-test evaluated as described in Section 3.3. In the parenthesis next to it, it is presented the fraction of networks for which the p-value was lower than 0.05.

t	Average	Standard deviation	Skewness	p-val (fraction < 0.05)
$t_{\bar{C}}$	$-193.127920^{+95.366190}_{-235.414020}$	160.654646	−3.246060	0.000581 (99.814%)
$t_{\bar{C}}$	$-330.674787^{+181.985206}_{-645.909165}$	397.203246	−9.187743	0.000001 (100.000%)
$t_{\bar{d}}$	$-190.808070^{+80.353118}_{-173.612148}$	123.645141	−2.145209	0.000098 (99.907%)
t_A	$-47.170522^{+23.227656}_{-57.537909}$	39.497893	−3.827625	0.000728 (99.814%)
t_{C_3}	$-333.068414^{+184.083597}_{-650.350176}$	399.742838	−9.028683	0.000217 (99.907%)

Our analysis did not focus in any given specific branch of the tree of life, such that the networks vary a lot in size. The average shortest path for both, metabolic and PPI networks, tend to be slightly smaller in the case of real graphs than the expected values in randomized networks with high statistical significance. For other parameters, while the metabolic networks also show some differences, these differences do not present the same confidence level significance over the whole sample. But in the case of PPI networks, all parameters analyzed do show differences between real and expected values with a high confidence level. PPI networks are more assortative, and have bigger local and global clustering coefficients than would be expected in a randomly picked (evolved) graph with the same degree distribution.
This study points to two important conclusions: on one hand, the degree distributions of graphs representing biological systems (metabolic and PPI networks) is not well fitted by a power-law function and on the other hand, network evolutionary models that focus solely in obtaining graphs with similar degree distributions as real world biological networks might not be enough to adjust other graph parameters observed in these systems.

Appendix. Parameter distribution over the randomized networks

In this work we compared the graph parameters of a real world network with their expected values in the set of all graphs that can be built with the same degree distribution. This analysis is inspired by the fact that many models that try to describe the evolution of these systems usually focus in obtaining as a result of the model simulation graphs with similar degree distributions. We ask, therefore, if mimicking this degree distribution is enough to reproduce the main properties of these structures. Given a degree distribution, the number of different graphs that can be actually built from it is astronomically huge and a random model adjusted only to reproduce a given degree distribution would, in principle, generate any of the possible networks that share this same distribution. So the question is: Do the characteristics of a real world network significantly differ from those of a randomly selected graph from the population of all networks that share the same degree distribution?
The process through which we obtain samples from this population is the randomization process described in Section 3.3, where links are broken and rewired in order to obtain a complete new network from the original but keeping the degrees of the nodes untouched. Though this rewiring process is computationally fast, the computation of all parameters of the resulting network, in particular P_2, C_3 and d_{ij}, is a computationally intense process and in order to perform all calculations in a relatively short and reasonable time period we chose to obtain a small random sample. Therefore, we had to use a statistical test designed to provide reliable results even for small samples: the student's t-test. One of the suppositions behind the student's t-test performed in the analysis is that the population behind the obtained sample has a normal distribution.

Table 11
Normality test for a few selected organisms. The average value of the parameters over the samples are shown and the two values next to each are the p-value of the Shapiro–Wilk test (pv1) and the p-value for the D'agostino test (pv2).

Organism	N	$\mathcal{N}$	Sample size	$\bar{C}$ (pv1, pv2)	A (pv1, pv2)	$\bar{d}$ (pv1, pv2)	c (pv1, pv2)
mgl	823	2 416	500	0.190693 (0.458, 0.510)	−0.230871 (0.016, 0.018)	3.020923 (0.483, 0.799)	0.054189 (0.170, 0.661)
syn	967	2 651	500	0.187642 (0.108, 0.138)	−0.211546 (0.350, 0.406)	3.058312 (0.510, 0.365)	0.041776 (0.482, 0.715)
eco	1195	3 564	500	0.192970 (0.399, 0.403)	−0.204386 (0.406, 0.227)	3.010009 (0.245, 0.145)	0.037497 (0.405, 0.275)
243 273	436	6 107	500	0.137612 (0.290, 0.216)	−0.046098 (0.134, 0.165)	2.200834 (0.124, 0.069)	0.132851 (0.866, 0.860)
511 145	4076	70 424	100	0.041745 (0.208, 0.256)	−0.044631 (0.271, 0.586)	2.639252 (0.315, 0.697)	0.033799 (0.470, 0.410)
4932	6018	252 957	100	0.046844 (0.428, 0.597)	−0.037580 (0.075, 0.181)	2.300632 (0.717, 0.538)	0.041193 (0.449, 0.342)

Here, for a few organisms, we obtained bigger random samples and performed two normality tests to check whether these populations comply with the needed supposition of normality in order to perform the t-test. In Table 11 we present the results for the PPI and metabolic networks of a few selected organisms. The table shows the average value of the given parameters in a sample of randomized networks and for each parameter two p-values next to it that are the result of the Shapiro–Wilk [35] and D'Agostino [36] normality tests. In any of the tests, a big p-value (bigger than the critical value) indicates that the sample seems to have a normal distribution. In the tables, for the PPI networks we identify the organisms by their NCBI taxonomy ID and metabolic networks by their KEGG code. Again, in this evaluation we reduced the sample sizes for those networks in which the calculations took longer time.

References

[1] P. Erdős, A. Rényi, On random graphs i., Publ. Math. (Debrecen) 6 (1959) 290–297.
[2] D.J. de Solla Price, Networks of scientific papers, Science 149 (3683) (1965) 510–515.
[3] N. Przulj, D. Wigle, I. Jurisica, Functional topology in a network of protein interactions, Bioinformatics 20 (3) (2004) 340–348.
[4] M.M. Babu, N.M. Luscombe, L. Aravind, M. Gerstein, S.A. Teichmann, Structure and evolution of transcriptional regulatory networks, Curr. Opin. Struct. Biol. 14 (3) (2004) 283–291.
[5] C. Li-Ping, W. Ru, S. Hang, X. Xin-Ping, Z. Jin-Song, L. Wei, C. Xu, Structural properties of US flight network, Chin. Phys. Lett. 20 (8) (2003) 1393.
[6] L.A.N. Amaral, A. Scala, M. Barthelemy, H.E. Stanley, Classes of small-world networks, Proc. Natl. Acad. Sci. 97 (21) (2000) 11149–11152.
[7] A.L. Barabasi, Scale-free networks: a decade and beyond, Science 325 (5939) (2009) 412–413.
[8] A.L. Barabasi, E. Bonabeau, Scale-free networks, Sci. Am. 288 (5) (2003) 60–69.
[9] R. Albert, Scale-free networks in cell biology, J. Cell Sci. 118 (Pt 21) (2005) 4947–4957.
[10] E. Ravasz, A.-L. Barabási, Hierarchical organization in complex networks, Phys. Rev. E 67 (2) (2003).
[11] R. Albert, H. Jeong, A.-L. Barabási, Error and attack tolerance of complex networks, Nature 406 (6794) (2000) 378–382.
[12] D.S. Callaway, M.E.J. Newman, S.H. Strogatz, D.J. Watts, Network robustness and fragility: Percolation on random graphs, Phys. Rev. Lett. 85 (2000) 5468–5471.
[13] R. Cohen, K. Erez, D. ben Avraham, S. Havlin, Resilience of the internet to random breakdowns, Phys. Rev. Lett. 85 (2000) 4626–4628.
[14] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[15] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks, Rev. Modern Phys. 74 (2002) 47–97.
[16] P.L. Krapivsky, S. Redner, Organization of growing random networks, Phys. Rev. E 63 (6) (2001).
[17] A. Vázquez, Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations, Phys. Rev. E 67 (2003) 056104.
[18] R. Khanin, E. Wit, How scale-free are biological networks, J. Comput. Biol. 13 (3) (2006) 810–818.
[19] G. Lima-Mendez, J. van Helden, The powerful law of the power law and other myths in network biology, Mol. BioSyst. 5 (2009) 1482–1493.
[20] A. Clauset, C.R. Shalizi, M.E.J. Newman, Power-law distributions in empirical data, SIAM Rev. 51 (4) (2009) 661–703.
[21] A.D. Broido, A. Clauset, Scale-free networks are rare, Nature Commun. 10 (1) (2019).
[22] I. Voitalov, P. van der Hoorn, R. van der Hofstad, D. Krioukov, Scale-Free Networks Well Done, 2018.
[23] V. Sudbrack, L.G. Brunnet, R.M. de Almeida, R.M. Ferreira, D. Gamermann, Master equation for the degree distribution of a duplication and divergence network, Physica A 509 (2018) 588–598.
[24] M. Kanehisa, S. Goto, KEGG: kyoto encyclopedia of genes and genomes, Nucleic Acids Res. 28 (1) (2000) 27–30.
[25] M. Kanehisa, Y. Sato, M. Kawashima, M. Furumichi, M. Tanabe, Kegg as a reference resource for gene and protein annotation, Nucleic Acids Res. 44 (D1) (2016) D457–D462.
[26] R. Reyes, D. Gamermann, A. Montagud, D. Fuente, J. Triana, J.F. Urchueguia, P.F. de Cordoba, Automation on the generation of genome-scale metabolic models, J. Comput. Biol. 19 (12) (2012) 1295–1306.
[27] D. Szklarczyk, A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K.P. Tsafou, M. Kuhn, P. Bork, L.J. Jensen, C. von Mering, STRING V10: protein-protein interaction networks, integrated over the tree of life, Nucleic Acids Res. 43 (Database issue) (2015) D447–452.
[28] E. Estrada, The Structure of Complex Networks: Theory and Applications, Oxford University Press, 2011.
[29] J.R. Fonseca, M.I. Friswell, J.E. Mottershead, A.W. Lees, Uncertainty identification by the maximum likelihood method, J. Sound Vib. 288 (3) (2005) 587–599, Uncertainty in structural dynamics.
[30] B.M. Hill, A simple general approach to inference about the tail of a distribution, Ann. Statist. 3 (5) (1975) 1163–1174.
[31] A.L.M. Dekkers, J.H.J. Einmahl, L.D. Haan, A moment estimator for the index of an extreme-value distribution, Ann. Statist. 17 (4) (1989) 1833–1855.
[32] P. de Wolf, H. Lopuha, P. Groeneboom, Kernel-type estimators for the extreme value index, Ann. Statist. 31 (6) (2003) 1956–1995.
[33] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271.
[34] M.E.J. Newman, The structure and function of complex networks, SIAM Rev. 45 (2) (2003) 167–256.
[35] S.S. Shapiro, M.B. Wilk, An analysis of variance test for normality (complete samples), Biometrika 52 (3–4) (1965) 591–611.
[36] R.B.D. Agostino, Transformation to normality of the null distribution of g1, Biometrika 57 (3) (1970) 679–681.