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

METABOLISM ANALYSIS OF *Streptomyces leeuwenhoekii* C34 WITH A GENOME SCALE MODEL AND IDENTIFICATION OF BIOSYNTHETIC GENES OF SPECIALISED METABOLITES BY GENOME MINING

TESIS PARA OPTAR AL GRADO DE DOCTOR EN CIENCIAS DE LA INGENIERÍA
MENCIÓN INGENIERÍA QUÍMICA Y BIOTECNOLOGÍA

VALERIA ISABEL RAZMILIC NEIRA

PROFESORES GUÍAS:
JUAN A. ASENJO DE LEUZE
BARBARA ANDREWS FARROW

MIEMBROS DE LA COMISIÓN:
ZIOMARA GERDTZEN HAKIM
BERNARDO GONZÁLEZ OJEDA
ORIANA SALAZAR AGUIRRE

SANTIAGO, CHILE
2017

SUMMARY OF THE THESIS FOR THE DEGREE OF: Doctor in Engineering Sciences mention Chemical Engineering and Biotechnology
BY: Valeria Isabel Razmilic Neira
DATE: 6^{th} January, 2017
SUPERVISORS: Dr. Juan A. Asenjo and Dr. Barbara Andrews

METABOLISM ANALYSIS OF *Streptomyces leeuwenhoekii* C34 WITH A GENOME SCALE MODEL AND IDENTIFICATION OF BIOSYNTHETIC GENES OF SPECIALISED METABOLITES BY GENOME MINING

Streptomyces leeuwenhoekii C34 is a novel strain isolated from the Chaxa lagoon of the Atacama Desert, Chile. It produces specialised metabolites with antibiotic activity against methicillin resistant *Staph. aureus* (MRSA): chaxamycins and chaxalactins. A reliable genome sequence of *S. leeuwenhoekii* C34 was obtained using Illumina Miseq and PAC-bio RS II SMRT technologies. The genome was used to identify biosynthetic gene clusters (BGCs) for specialised metabolites through genome mining, and to develop a genome scale model (GSM) to study the biosynthesis pathways for production of specialised metabolites.

Thirthy-four BGCs were found in the genome of *S. leeuwenhoekii* C34, plus one BGC coded in the plasmid pSLE2. Three lasso-peptide BGCs were found. Specifically, the product of the lasso-peptide 3 BGC was identified in the supernatant of *S. leeuwenhoekii* C34 grown in TSB/YEME culture medium and it was successfully expressed in the heterologous host *S. coelicolor* M1152. It was confirmed that this lasso-peptide was the same as the chaxapeptin described for *S. leeuwenhoekii* C58. Also, a 64 kb BGC (locus 1083651 to 1147687) was identified encoding for a hybrid *trans*-AT PKS/NRPS, that probably produces a halogenated specialised metabolite due to the presence of the gene *sle09470* that encodes for a chlorinating enzyme. In order to study the hybrid *trans*-AT PKS/NRPS BGC, several strains derived from *S. leeuwenhoekii* C34 were developed. The BGC was also cloned in heterologous hosts *S. coelicolor* M1152, M1154, and *S. albus*. Through HPLC MS/MS analysis and metabolite profile comparison a group of compounds was identified with a chlorinated pattern, however, they were not exclusive to *S. leeuwenhoekii* strains since they were also observed in the supernatant of *S. coelicolor* M1152, so they were rejected as possible products of the hybrid *trans*-AT PKS/NRPS BGC. On the other hand, a metabolite with a *m/z* 611.53 $[M + H]^+$ signal was detected only in the strains *S. leeuwenhoekii* M1614 (Δchaxamycin BGC) and M1619 (Δchaxamycin BGC; Δ*sle09560*). Further studies are needed in order to confirm if the differently expressed metabolite corresponds to a product of the hybrid *trans*-AT PKS/NRPS BGC.

For the development of the GSM of *S. leeuwenhoekii* C34, a python based interface was developed, which enables: a search for the *Streptomyces* genes associated with a reaction in the KEGG database, performance of local BLAST against *S. leeuwenhoekii* C34, comparison of the domain of the proteins, downloading of the metabolite information, construction of the GSM and performance of flux balance analysis (FBA) simulations using COBRApy. The biosynthesis pathways of specialised metabolites such as chaxamycins, chaxalactins, desferrioxamines, ectoine and the product of a hybrid *trans*-AT PKS/NRPS BGC (hybrid PK-NP) were included in the model. The model, *i*VR1007, consists of 1722 reactions, 1463 metabolites and 1007 genes, and it was validated using experimental information of growth in different carbon, nitrogen, and phosphorous sources, showing a 83.7 % accuracy. Within the model, non-intuitive gene knockouts and gene overexpressions that predicted an enhanced flux through the production of building blocks of chaxamycins, chaxalactins and the hybrid PK-NP, were found. The predicted modifications would be used to perform metabolic engineering of *S. leeuwenhoekii* C34 in order to increase specialised metabolite production.

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE: Doctor en Ciencias de la Ingeniería mención en Ingeniería Química y Biotecnología
POR: Valeria Isabel Razmilic Neira
FECHA: 6 de Enero de 2017
PROF. GUÍA: Dr. Juan A. Asenjo y Dra. Barbara Andrews

ANÁLISIS DEL METABOLISMO DE *Streptomyces leeuwenhoekii* C34 A TRAVÉS DE UN MODELO A ESCALA GENÓMICA E IDENTIFICACIÓN DE GENES BIOSINTÉTICOS DE NUEVOS METABOLITOS ESPECIALIZADOS MEDIANTE MINERÍA DE GENOMAS

Streptomyces leeuwenhoekii C34 es una nueva cepa que fue aislada desde la laguna Chaxa ubicada en el Desierto de Atacama, Chile. Esta cepa produce metabolitos especializados con actividad contra *Staph. aureus* resistente a meticilina (MRSA): chaxamicinas y chaxalactinas. La secuencia genómica de *S. leeuwenhoekii* C34 se obtuvo mediante las tecnologías de Illumina Miseq y PACbio RS II SMRT. El genoma se utilizó para identificar clústers de genes biosintéticos (BGCs) que codifican para metabolitos especializados a través de minería de genomas, y para desarrollar un modelo a escala genómica (GSM) para estudiar las rutas de biosíntesis de producción de metabolitos especializados.

Se encontraron 34 BGCs en el genoma de *S. leeuwenhoekii* C34, más un BGC ubicado en el plásmido pSLE2. Se encontró tres BGCs para lazo-péptidos. Específicamente, se identificó el producto del BGC del lazo-péptido 3 en el sobrenadante de *S. leeuwenhoekii* C34 cultivado en medio TSB/YEME y se expresó exitosamente en el huésped heterólogo *S. coelicolor* M1152. Se confirmó que este lazo-péptido era el mismo que la chaxapeptina, recientemente descrita para *S. leeuwenhoekii* C58. Por otra parte, se identificó un BGC de 64 kb (locus 1083651 a 1147687) que codifica para un híbrido *trans*-AT PKS/NRPS. Es probable que el producto de este BGC sea un compuesto halogenado debido a la presencia de un gen, *sle09470*, que codifica para una enzima cloradora. Para estudiar este clúster de genes, se desarrollaron diferentes cepas derivadas de *S. leeuwenhoekii*. También, el BGC se clonó en huéspedes heterólogos: *S. coelicolor* M1152, M1154 and *S. albus*. A través de análisis de HPLC MS/MS y comparación de perfiles de metabolitos, se identificó un grupo de compuestos con patrón clorado, sin embargo se descartaron como posibles productos del BGC ya que además de encontrarse en las cepas de *S. leeuwenhoekii* también se encontraron en muestras de *S. coelicolor* M1152. Por otra parte, se detecto un metabolito con una señal de *m/z* 611.53 $[M + H]^+$ solamente en las muestras de *S. leeuwenhoekii* M1614 (Δchaxamycin BGC) y M1619 (Δchaxamycin BGC; Δ*sle09560*). Se requieren más estudios para confirmar si los metabolitos expresados diferencialmente corresponden a un producto del híbrido transAT-PKS/NRPS BGC.

Para construir el GSM de *S. leeuwenhoekii* C34 se desarrolló una interfaz basada en python, que permite: buscar genes de *Streptomyces* asociados a reacciones en la base de datos KEGG, realizar BLAST local contra *S. leeuwenhoekii* C34, comparar los dominios de proteínas, descargar información de los metabolitos, construir el GSM y realizar simulaciones usando COBRApy. Las rutas biosintéticas de chaxamicinas, chaxalactinas, desferrioxaminas, ectoina y el producto del híbrido transAT-PKS/NRPS BGC (híbrido PK-NP) se incluyeron en el modelo. El modelo, *i*VR1007, consiste de 1722 reacciones, 1463 metabolitos y 1007 genes, y se validó usando información experimental de crecimiento en diferentes fuentes de carbono, nitrógeno y fósforo, mostrando un 83.7 % de precisión. El modelo se usó para encontrar deleción y sobre-expresión de genes no intuitivas que predicen un aumento en la producción de precursores de chaxamicinas, chaxalactinas e híbrido PK-NP. Las modificaciones predichas podrán ser usadas para realizar ingeniería metabólica de *S. leeuwenhoekii* C34 para incrementar la producción de metabolitos especializados.

To my wonderful parents
and my lovely husband

"Now, a living organism is nothing but a wonderful machine endowed with the most marvellous properties and set going by means of the most complex and delicate mechanism."
Claude Bernard

LIST OF PUBLICATIONS GENERATED FROM THIS WORK

V. Razmilic, J. F. Castro, J. P. Gomez-Escribano, M. J. Bibb, B. Andrews and J. A. Asenjo. Analysis of metabolic networks of *Streptomyces leeuwenhoekii* C34 by means of a genome scale model: prediction of modifications that enhance the production of specialized metabolites. In preparation.

J. P. Gomez-Escribano, J. F. Castro, **V. Razmilic**, G. Chandra, B. Andrews, J. A. Asenjo and M. J. Bibb (2015). *Streptomyces leeuwenhoekii* Genome: de novo Sequencing and Assembly in Single Contigs of Chromosome, Circular Plasmid pSLE1 and Linear Plasmid pSLE2. *BMC Genomics*, volume 16, issue 1, page 485. DOI 10.1186/s12864-015-1652-8

ACKNOWLEDGEMENTS

During the development of my post-graduate studies I had continued improving my scientific career and expanding my knowledge. I had passed through good and not so good moments and I had always felt supported and motivated to continue forward by the people around me. My sincere gratitude is for the following people.

Firstly, I would like to express my deep gratitude to my supervisors, Juan A. Asenjo and Barbara Andrews, for their constant support, assistance and guidance. They always motivate me to work hard and they gave me the opportunity to work in this wonderful project that involved working with a very interesting microorganism that was isolated from the Atacama Desert. Besides my supervisors, I would like to thank the rest of my thesis committee for their insightful comments and hard questions.

I would also like to express my sincere appreciation to Mervyn Bibb, who received me in his laboratory at John Innes Centre, Norwich, United Kingdom. He gave me excellent advice and guidance during my internship. Also, he was always willing to help and Merv contributed a lot to the development of my thesis. See you later, alligator.

I wish to thank the help provided by Juan Pablo Gomez-Escribano, aka JuanPa, who teach me the genetic procedures for modifying *Streptomyces* strains at John Innes Centre. Also, for his friendship and infinite patience to answer my doubts and questions.

I also thank Jean for his friendship, advice, and support, through all these years. I would like to extend my gratitude to Emilie and my colleagues that are part of the ”GBA” group for the encouragement and helpful discussions. In addition, I want to thank all the people that integrate the Centre of Biotechnology and Bioengineering (CeBiB) of the University of Chile, for generating a great work-environment and especially for the advice given during seminars.

I am grateful of my fellow labmates of the Department of Molecular Microbiology of the John Innes Centre, that help me by sharing their expertise in the genetic manipulation of *Streptomyces*, and also, for all the fun that I had during the internship.

My special thanks to Alan Bull and Michael Goodfellow who had always shown me their love for science, and had always shared their knowledge about microbiology and taxonomy, especially in the isolation of strains from extreme environments and in the taxonomic classification of the species. Since I started with my P.hD. project they had always being able to give good advice and to encourage me to develop my thesis.

I would like to express my sincere gratitude to my family, especially to my parents, that have always believe in me, and had shown me the life of science since I was a kid.

Particularly, I want to thank my husband, Rubén, who has been next to me during all my P.hD.

studies (and before that) and had encouraged me to pursue my dreams, for his unconditional love and support.

Finally, I thank the National Commission for Scientific and Technological Research (CONICYT) for the National Doctorate Fellowship (#21110384, 2011–2015) and Operating Expenses (2013 and 2014). I would like to thank the Assistance to Events and Short Courses for Doctorate Students Grants (#81140192, 2014) and the Becas Chile Programme of CONICYT: Doctoral Internship Abroad Scholarship (#75130080, 2013–2014). Also, I want to thank the Basal Programme of CONICYT (#FB0001) and the Newton project: Bioprospecting the Atacama Desert: the discovery and enhancement of novel therapeutic drugs from actinomycetes (JIC #CA586).

Contents

List of Tables

List of Figures

Introduction

Specialised metabolites and microbial metabolism

Specialised or secondary metabolites are structurally diverse small molecules synthesised by a wide range of microorganisms (bacteria and fungi) and organisms (plants and animals). Although they had been detected in some animals later studies have suggested that there are being produced by symbiotic microorganisms (Kobayashi and Ishibashi, 1993). They accomplish diverse roles within the microbial environment such as defence, communication, metal transporters, sexual hormones and as differentiation effectors (Demain and Fang, 2000). Specialised metabolites could be classified into five main classes: alkaloids, terpenoids, fatty acid-derived and polyketides, non-ribosomal peptides, and enzyme cofactors (McMurry and Begley, 2005).

Specialised metabolites have shown to have diverse chemical structures and many different applications, especially in human medicine. It is usual that specialised metabolites are produced in discrete amounts by the wild-type strains, so there is a need to improve their production to accomplish practical requirements (Olano et al., 2008).

Understanding how specialised metabolites are synthesised by a microorganism is essential for improving their production, and for doing so is important to understand their metabolism. The metabolism represents the connection of biochemical reactions that allow an organism, like bacteria, to grow and have particular characteristics. The metabolism of bacteria could be divided into primary and secondary stages/phases. The primary metabolism is common to all the microorganisms and is integrated by all the essential reactions that allow growth. During this metabolism the microorganism is growing exponentially. On the other hand, the secondary metabolism includes species-specifics pathways and occurs during the stationary phase (Hiltner et al., 2015). In the course of exponential growth precursors for cell growth are synthesized and some of them are going to be used during the secondary metabolism to produce specialised metabolites that will confer a distinct characteristic to the strain (such as higher likelihood of survival). Specialised metabolites are mainly, but not exclusively, being synthesized during the secondary metabolism and strongly depend on the precursors synthesised in the primary metabolism (Hiltner et al., 2015).

On the other hand, specialised metabolites are usually encoded in biosynthetic gene cluster (BGC) in the chromosome or in extra-chromosomal DNA of microorganisms. The BGCs include the biosynthetic genes, and may also incorporate regulatory, transport, resistance or post-synthetic

modification genes. The expression of specific pathway regulators frequently depends on genes that are required for the production of several metabolites produced by the strain (Bibb, 2005). The size of the BGC can be from just a few to over 100 kb (Ōmura et al., 2001; Bentley et al., 2002; Gomez-Escribano et al., 2015).

Why is it important to find new specialised metabolites?

Specialised metabolites are the main source of bioactive compounds that could have broad uses, for example in agriculture, forestry, and in medicine. About 70 % of anti-infectives (anti-bacterial, -fungal, -parasitic and -viral) and 63 % of anticancer agents, are naturally derived or inspired (Cragg and Newman, 2009).

The first antibiotic (anti-bacterial) was discovered by Alexander Fleming in 1928, the penicillin (Fleming, 1929). He observed that a contamination on his plate, the mold *Penicillium notatum*, had an inhibitory effect over *Staphylococcus aureus*. Later, penicillin was developed to a medicine by Florey and Chain in the 40s. Since then a brief golden era of antibiotics began (see Figure 1), while in later years there have been just a few new classes of antibiotics discovered. The decrease in major new antibiotics discovery in conjunction with the apparition of multidrug resistance strains due to overuse, miss-use of antibiotics, horizontal gene transfers and spontaneous resistant mutants has become a world-wide problem. Specifically in Chile, there has been an increasing tendency in the percentage of resistant clinical isolates (considering from 1991 to 2014), with the exception of resistance to gentamicin by *Pseudomonas aeruginosa* that had decreased (Figure 2).

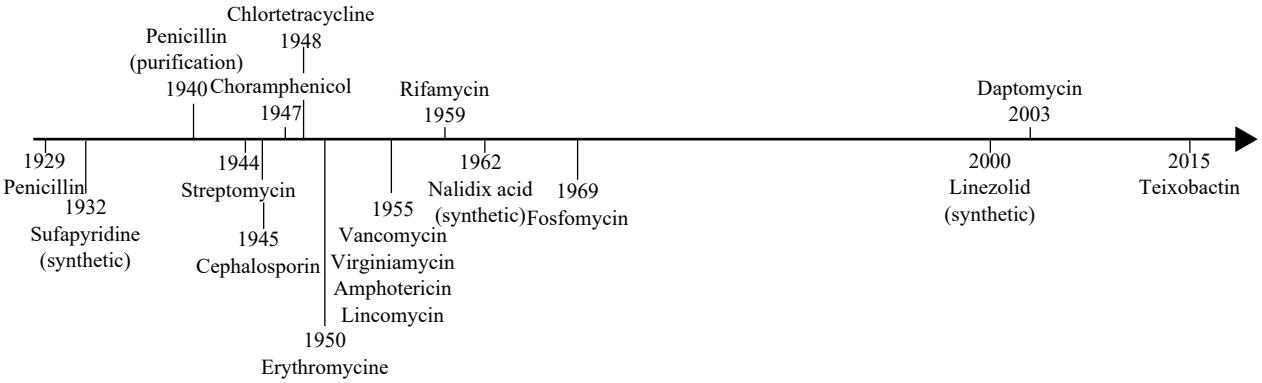


Figure 1: Time-line of discovery of new antibiotic classes. Based on information from (Walsh and Wright, 2005).

In the meantime, it was realized that compounds which possess antibiotic activity also possess other activities and they were starting to be used to treat other kinds of infectious diseases and to treat diseases like cancer that previously were treated just with synthetic compounds (Demain, 1999). Antitumor antibiotics are amongst the most important chemotherapeutic agents to treat cancer, and with the exception of semi-synthetic compounds, they were isolated from *Streptomyces* strains (Cragg and Newman, 2009). Some specialised metabolites with antitumor activity are daunomycin (Kersten and Kersten, 1965), chromomycin A_3 (Kersten and Kersten, 1965; Menéndez et al., 2004), leinamycin (Hara et al., 1989, 1990; Gates, 2000) and bleomycin (Blum et al., 1973).

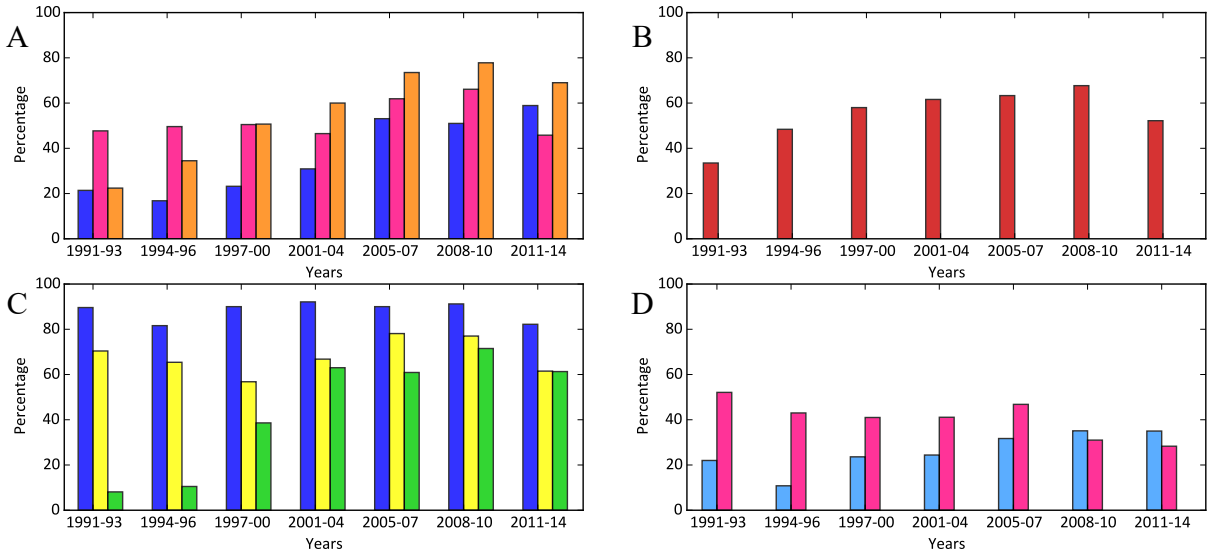


Figure 2: Percentage of resistance to antibiotics of Chilean clinical isolates from 1991 until 2015* of, *Klebsiella pneumoniae* **A**, *Staphylococcus aureus* **B**, *Acinetobacter baumannii* **C**, and *Pseudomonas aeruginosa* **D**. Blue: Ciprofloxacin, pink: Gentamicin, orange: Cefotaxime, red: Cloxacillin, yellow: Amikacin, green: Ampicillin/sulbactam, light blue: Ceftazidime. Data were taken from a report of the Public Health Institute of Chile (available at http://www.ispch.cl/sites/default/files/BoletinRam-30112015A_0.pdf). *In 2010 no data was collected.

There is an urgent need to find new specialised metabolites to overcome the issue of multidrug resistant strains, and also to exploit the potential of specialised metabolites to be therapeutic agents to treat other diseases, like cancer.

Streptomyces as a source of specialised metabolites

Streptomyces are soil Gram-positive bacteria that are characterized for developing a mycelial mass, having a high G+C content and large genomes (more than 7 Mbp) (Kieser et al., 2000).

The life cycle of *Streptomyces* starts when one or two germ tubes emerge from a spore under favourable conditions, then vegetative mycelium is developed and subsequently, aerial hyphae are formed ending with hyphae segmentation that leads to spore formation (Kieser et al., 2000). This particular life cycle involves the establishment of the colony in a niche and that is why it has been proposed as one of the reasons why this microorganism produces several compounds.

The majority of specialised metabolites are isolated from bacteria and fungus. Among the bacteria genera, 2/3 of the known metabolites with activity have been isolated from actinomycetes and 80% of those from members of the genus *Streptomyces* (Kieser et al., 2000).

Streptomyces leeuwenhoekii

Streptomyces leeuwenhoekii C34, formerly *Streptomyces* sp. C34, is a bacterial strain isolated from the Chaxa lagoon located in the Atacama Desert, Chile (Okoro et al., 2009). This strain is characterised for the production of novel antibiotics with activity against methicillin resistant *Staphylococcus aureus* (MRSA), the Chaxalactins and Chaxamycins, the latter also haves antitumor activity (Rateb et al., 2011a,b). Chaxamycins and chaxalactins correspond to polyketides according to their structures and BGCs. This bacterium also produces known compounds, desferrioxamine E, and hygromycin A. A first genome sequence of *S. leeuwenhoekii* was achieved using Illumina technology (Busarakam et al., 2014). Assembly of the 658 contigs obtained gave a genome size of 7.86 Mb, with 6,780 predicted open reading frames, and a G+C content of 72.6 % (Busarakam et al., 2014). Among the phenotypic properties of *S. leeuwenhoekii*, it has been described that it forms spiral spore chains, it has a gray-yellow-green aerial spore mass, a gray-yellow substrate mycelium and a pale-yellow diffusible pigment, and it is able to grow between 10 and 50 °C, at pH 10 and at 7.5 % w/v NaCl (Busarakam et al., 2014). The BGC that encoded for chaxamycins had been identified and studied previously and there is a proposed biosynthetic pathway (Castro et al., 2015). The chaxalactins BGC is currently being studied (Castro et al., in preparation).

Polyketides

Polyketides are a relevant group of specialised metabolites that are synthesised by polyketide synthase (PKS). There are three types of PKS. First, type I PKSs are multifunctional enzymes that are organized into modules, each of which includes a set of non-iteratively acting domains responsible for one cycle of polyketide chain elongation. Second, type II PKSs are multienzyme complexes that carry a single set of iterative domains. Third, type III PKSs, also known as chalcone synthase-like PKSs, are homodimeric enzymes that contain iteratively-acting ketosynthase (KS) domains (Shen, 2003).

From now on, when speaking of PKS it will be referring to type I PKS unless otherwise is stated. Each module of the PKS contains three essential domains; acyltransferase (AT), KS, and acyl-carrier protein (ACP). In addition, other domains could be present in the modules such as ketoreductase (KR), enoyl reductase (ER), dehydratase (DH), methyl-transferase (MT), among others (Figure 3). From the domain configuration it is possible to predict the product of the BGC, taking into consideration the fingerprints that are present in each domain.

The AT domain recognizes and transfers the acyl molecules that would be incorporated into the polyketide chain. Which starter and extender units will be chosen by the AT domain could be predicted based on fingerprint residues present in the amino-acid sequence: the motif YASHS is associated with recognition of methyl-malonyl-CoA, while HAFHS is associated to malonyl-CoA (Keatinge-Clay, 2012). The KS domain performs the Claisen condensation reaction between the polyketide chain and the extender unit. Specific amino acid residues indicate if this domain is active. The conserved motifs of KS are: Tx**C**xxS, **H**GTGT and GSVKxxxG**H**, the catalytic triad residues are shown in bold. The ACP domain contains a flexible 4'-phosphopantetheine prosthetic group added post-translationally to the serine of the (D/E)xGxD**S** motif. This domain shuttles

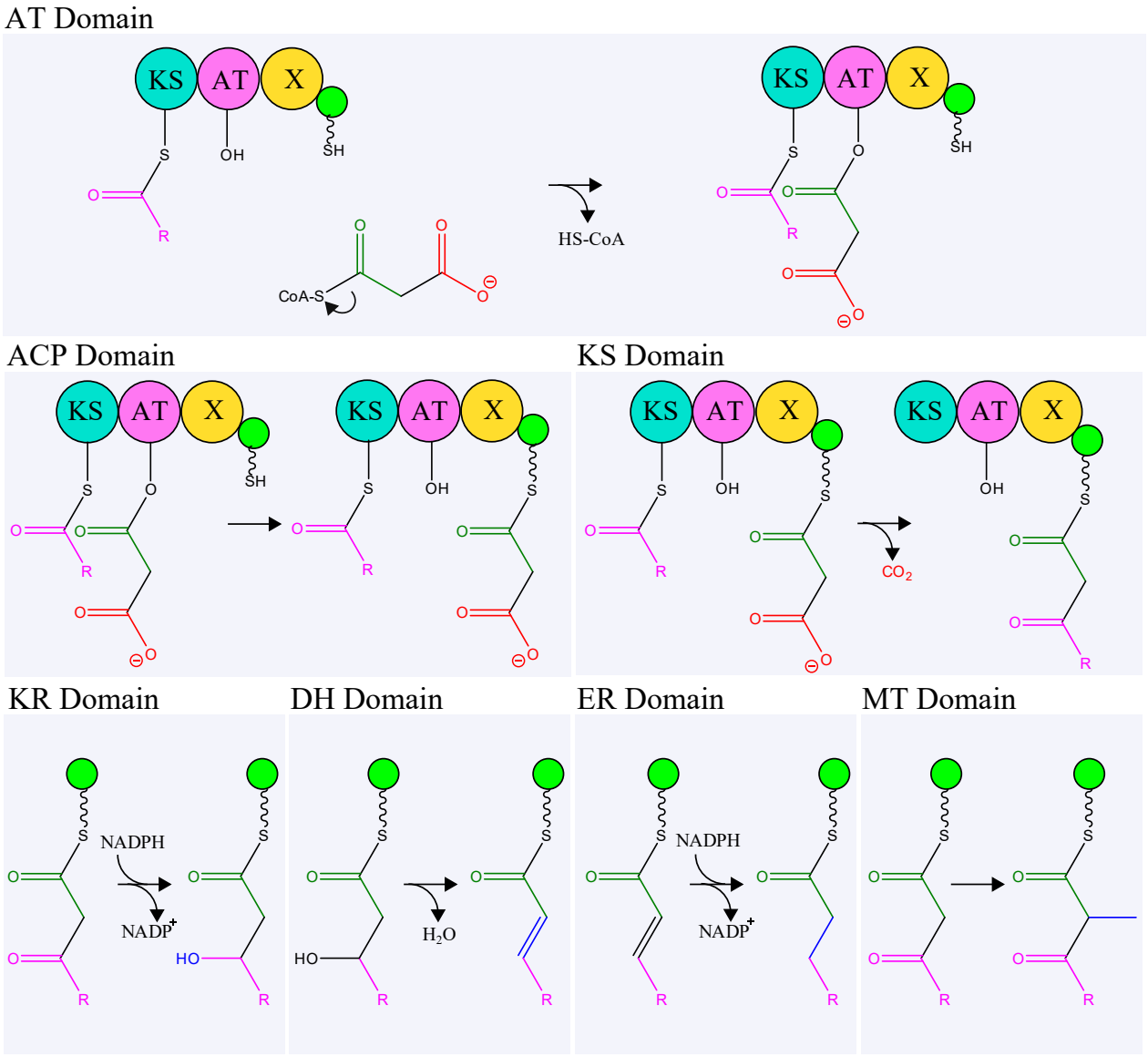


Figure 3: Reactions catalyzed by PKS domains. Green: ACP domains; X: represents additional modification domains like the ones showed at the bottom of the figure. Adapted from (Nguyen et al., 2008).

extender units and polyketide intermediates through the thioester linkage (Keatinge-Clay, 2012).

The KR domain reduces the β-keto group of a β-ketoacyl intermediate formed by the KS domain. During this process, the KR utilizes NADPH and sets the stereochemistry of the β-hydroxyl group. Accordingly to the arranged stereochemistry, KRs are classified as: A-type KRs which generate a hydroxyl group with an L-orientation, B-type KRs which generate a hydroxyl group with a D-orientation, and C-type KRs which lacks the ability to reduce polyketides but may possess an epimerase activity. Furthermore, a KR can operate on an α-substituted polyketide, in which case to the A-type and B-type KRs are annexed a "1" or a "2" if the α-substituent is in a D-orientation or in an L-orientation, respectively (Caffrey, 2003).

The DH domain catalyzes the dehydration of a polyketide intermediate to produce a double bond between the α and β carbons and removing a hydroxyl group. The conserved motifs of

DH are **D**(A/V)(V/A)(A/L)(Q/H) and **H**xxxGxxxxP, catalytic residues are shown in bold. The ER domain reduces the double bond generated by the DH domain. The conserved motif that is found in ER is GGVGxAAxQxA. The MT domain transfers a methyl group to the α-carbon of a β-ketoacyl intermediate, and are S-Adenosyl methionine (SAM)-dependent. The SAM-binding site of functional MT has the motif A(D/E)xGxGxG followed with a D or E $\sim$20 amino acids downstream, however, variations in this motif are common (Keatinge-Clay, 2012).

It should be noted that the AT domain could be absent in the module configuration, and instead a trans-acting AT domain is coded in another gene within the BGC, this type of polyketide are called *trans*-AT polyketides, for more details of this type of PKS go to Section: Hybrid non-ribosomal peptide/*trans*-AT polyketide.

Non-ribosomal peptides

Non-ribosomal peptides are synthesised by non-ribosomal peptide synthetase (NRPS). NRPSs are modular enzymes that use amino acids as building blocks. A minimal module configuration includes three domains: condensation (C), adenylation (A) and peptidyl-carrier protein (PCP). The reactions catalysed by frequently found domains on NRPS are summarized in Figure 4.

The A domain, selects and activates the amino acids that would be incorporated into the non-ribosomal peptide by adenylation. The active site contains a code of 10 amino acids that consist of a conserved lysine and an aspartate, three hydrophobic residues (in most cases), and five variable residues that control substrate specificity (Stachelhaus et al., 1999). The A domains can have relaxed substrate specificities (Marahiel and Essen, 2009). For prediction of the specificity there are several on-line tools that can be used like LSI based A-domain predictor (Baranašić et al., 2014) (http://bioserv7.bioinfo.pbf.hr/LSIpredictor/AdomainPrediction.jsp) and NRPSpredictor2 (Rausch et al., 2005; Röttig et al., 2011) (http://nrps.informatik.uni-tuebingen.de).

Then, similarly to the ACP domain, the PCP domain has a 4-phosphopantetheinyl arm attached post-translationally to the conserved serine of the motif GGX**S**, to which the adenylate reacts and forms an activated thioester derivative. The C domain catalyzes the peptide bond formation between the aminoacyl thioester attached to the PCP and the non-ribosomal peptide chain. The C domain can by replaced by a condensation/cyclization (Cy) domain, which catalyzes the condensation and the heterocyclisation of serine, cysteine or threonine (Challis and Naismith, 2004). Other domains, that could be found in NRPS are oxidation (Ox) or MT.

Hybrid non-ribosomal peptide/*trans*-AT polyketide

Hybrid non-ribosomal peptide/polyketides are compounds that are synthesised by an integrated system of NRPS and PKS that mediates direct transfer of a NRPS-bound peptidyl intermediate to a PKS module or vice versa (Du et al., 2001). So this type of compounds uses amino acids and acyl-CoA molecules as building blocks. A *trans*-AT polyketide is synthesised by a PKS that is missing the AT domain in each module, and instead there is a gene that encodes for the AT domain

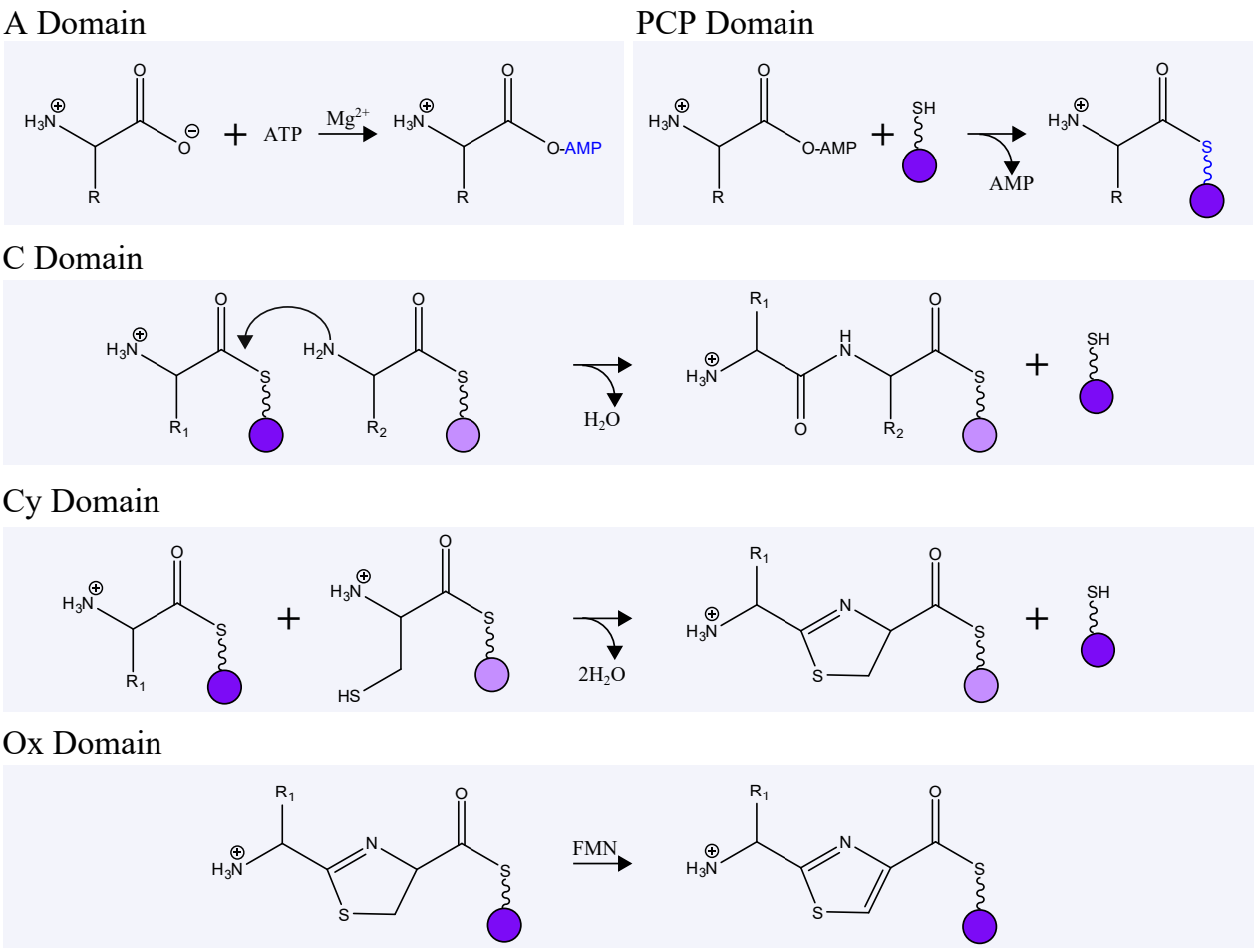


Figure 4: Reactions catalyzed by most common NRPS domains. Purple: PCP domains. Adapted from (Finking and Marahiel, 2004) and (Walsh et al., 2001).

in another part of the BGC.

Trans-AT polyketides usually have different characteristics such as domain composition (it is normal to find repetitive domains like more than one ACP), it can also be subject to further modification during the biosynthesis like β-branching (Helfrich and Piel, 2016). β-branching is a modification that consists of several steps and involves several genes: ACP, 3-hydroxy-3-methylglutaryl-CoA synthases (HMGS), enoyl-CoA dehydratases (ECH), a free standing non-elongating KS (KS*) (Figure 5). The first step in β-branching is the decarboxylation of malonyl-CoA by the free standing KS **1**, followed by aldol addition of the acetyl-ACP to the polyketide chain by an HMGS **2**, subsequently the aldol adduct is dehydrated and often decarboxylated by ECH **3** and **4** (Helfrich and Piel, 2016).

Trans-AT polyketides usually have inactive KS domains that lack the motif **H**GTGT (catalytic residue shown in bold), this type of domain (KS*) has been postulated to just pass the polyketide intermediate to the next module (El-Sayed et al., 2003).

Leinamycin, an antitumor antibiotic compound produced by *S. atroolivaceus*, was the first

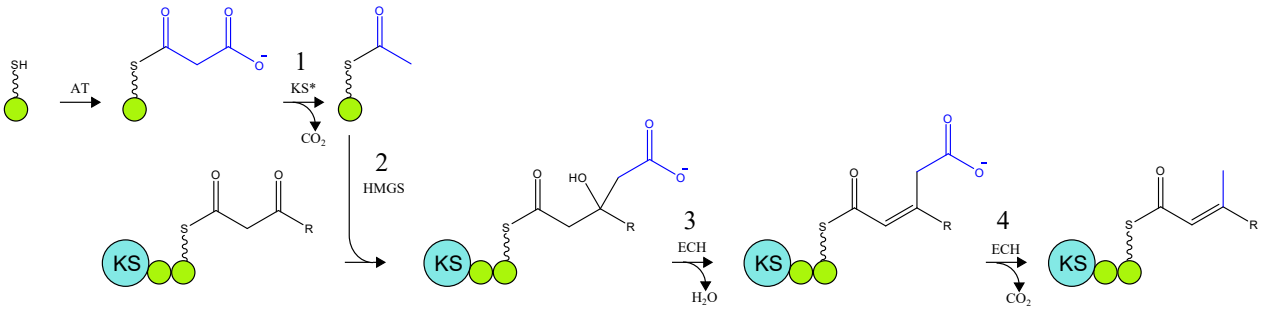


Figure 5: Representation of the general mechanism of β-branching (Helfrich and Piel, 2016). Green: ACP domains. For details see the text.

molecule found to be synthesised by a *trans*-AT PKS (Cheng et al., 2003). The domain configuration of the biosynthetic genes of the leinamycin BGC includes repeated ACP domains and other rare domains (Figure 6). The study of the usage of the ACP domains revealed a skipping mechanism, in which either ACP domain present in the module is able to carry on the polyketide intermediate to accomplish leinamycin production (Tang et al., 2006). The functionality of the distinct domains found in the biosynthetic genes and also of the β-branching genes will be further discussed in Chapter 1.

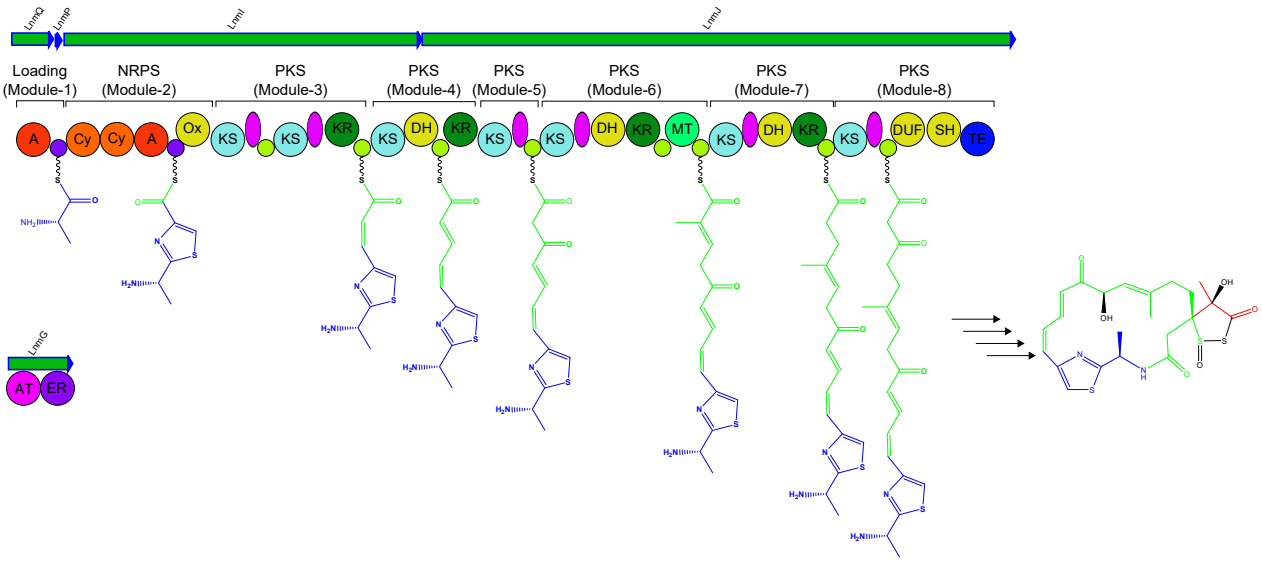


Figure 6: Leinamycin BGC as and example of hybrid *trans*-AT PKS/NRPS. Adapted from (Helfrich and Piel, 2016).

Whole genome sequencing

Since the first sequencing technique introduced by Sanger (Sanger and Coulson, 1975), several new techniques have been developed that allow obtaining genome sequences of diverse organisms with high speed, and can be used to get high quality sequences. This set of techniques is called next generation sequencing (NGS), among them Illumina MiSeq (http://www.illumina.com/) and Pacific Biosciences RSII (SMRT technology) (http://www.pacb.com/).

The sequenced genome of strains, like *S. coelicolor* (Bentley et al., 2002) or *S. avermitilis*

(Ōmura et al., 2001), have allowed the discovery of BGCs encoding for specialised metabolites that have not been detected experimentally (Lautru et al., 2005; Challis, 2008b). These BGCs are called cryptic or silent BGCs because they are not expressed under the studied conditions. The cryptic or silent BGCs encoded in the genome of diverse microorganisms offer a good chance for the discovery of new specialised metabolites such as antibiotics or anticancer compounds.

From the genome to new compounds

Genome mining is a technique that among other things is used for the discovery of new specialised metabolites (Lautru et al., 2005; Gross, 2007; Wilkinson and Micklefield, 2007; Challis, 2008a,b; Ikeda et al., 2014). The strategies followed in this work to try to identify new specialised metabolites are shown in Figure 7. First, an organism of interest is identified and selected, then within the genome sequence of the organism, and using bioinformatics tools to annotate the genome (e.g. RAST (Aziz et al., 2008)) and identify putative gene clusters (e.g. AntiSmash (Medema et al., 2011a; Blin et al., 2013; Weber et al., 2015)) groups of genes encoding for specialised metabolites are identified. Selected BGCs encoding for putative novel specialised metabolites are subjected to further approaches in order to identify/confirm the product of the BGC.

To study the metabolism: Genome scale models and flux balance analysis

Mathematical models are used to describe, study and understand the different behaviours of a diversity of systems. Genome scale models (GSMs) are reconstructions of the metabolism of organisms based on the genetic information available. Within these models and using flux balance analysis (FBA) it is possible to calculate the flow of metabolites through a network and predict the growth rate of an organism or the production rate of a relevant metabolite by maximization of an objective function (Orth et al., 2010).

The metabolic reactions of the GSM are defined in a stoichiometric matrix S of size $m \times n$, where m represents the number of metabolites and n the number of reactions. The entries in each column are the stoichiometric coefficients of the metabolites participating in a reaction in which consumed metabolites have a negative coefficient, produced metabolites have a positive coefficient and non-participating metabolites have a zero coefficient.

At steady state the system of linear equations of mass balance is given by:

$$Sv = 0 \tag{1}$$

where, S is a stoichiometric matrix and v is a vector of length n that represents the flux through all the reactions.

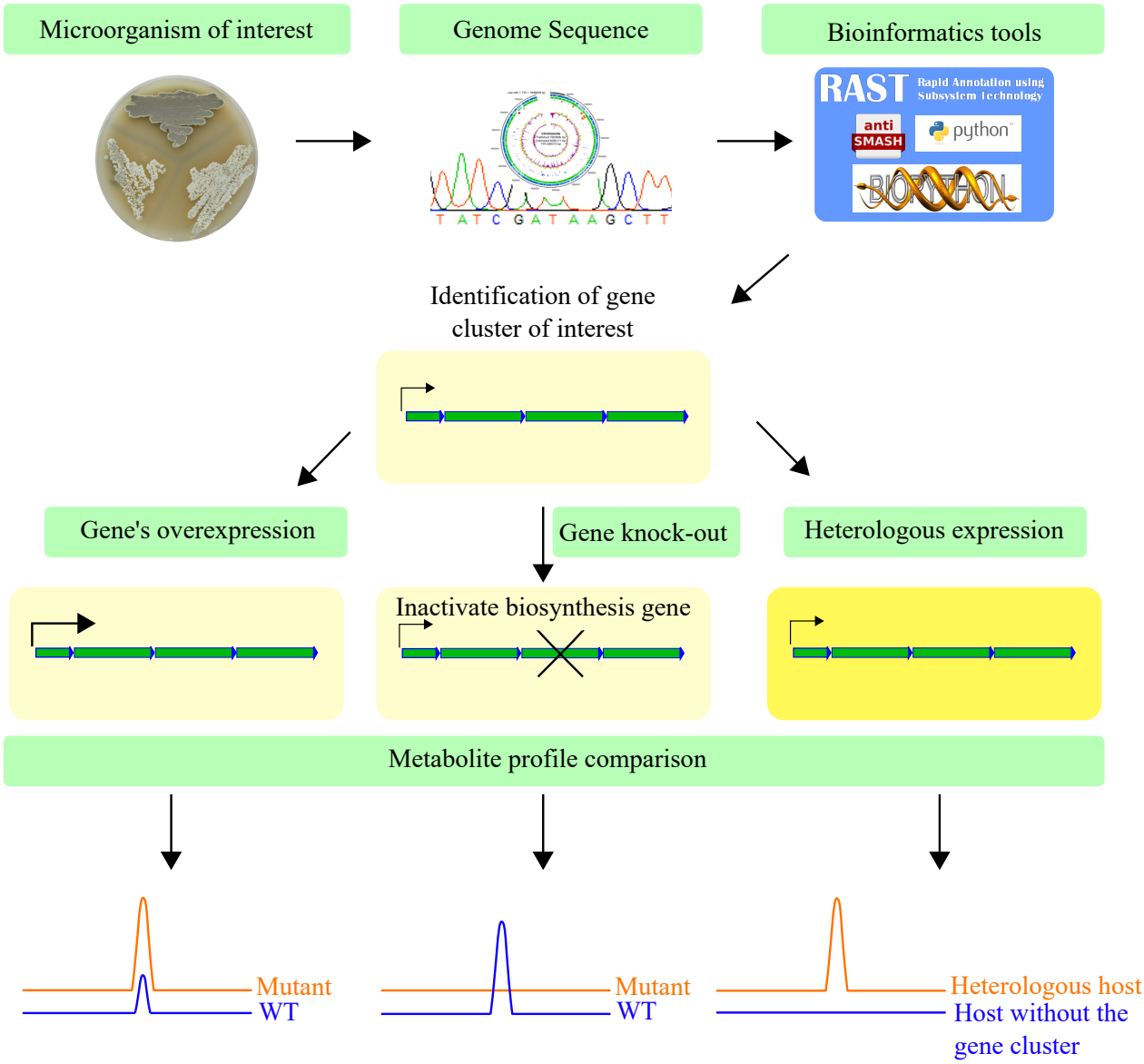


Figure 7: Methodology used to identify specialised metabolites products of silent BGC, through genome mining.

A solution space is defined by the constraints that are added to the model (lower bound and upper bound of each reaction). FBA maximizes or minimizes an objective function Z, given a solution space of a system. The outcome of the FBA is a particular flux distribution v. The objective function is defined by:

$$Z = c^T v \tag{2}$$

where c is a vector of weights indicating how much each reaction v contributes to the objective function. In practice, when only one reaction is desired for maximization or minimization a value of 1 is assigned at the position of the reaction of interest and a value of 0 is assigned to the rest of reactions.

The usual amount of specialised metabolites produced by the wild-type (WT) microorganism is discrete and several approaches have been used to accomplish a higher production of the metabolite of interest. FBA could be used to identify genes that could be deleted or overexpressed in order to improve the production of a specific metabolite.

Minimal of metabolic adjustment (MOMA) is an alternative technique to FBA to study perturbations in the metabolic network. MOMA uses the same constraints as FBA, but relaxes the assumption of optimal growth, taking into consideration that a gene deletion in the network will produce a minimal redistribution of the fluxes with respect to the WT (Segre et al., 2002). To solve a problem through MOMA it is necessary to employ quadratic programming to calculate the distance minimization in a flux space (Segre et al., 2002).

The use of either MOMA or FBA within a GSM has proven to be a powerful tool to study the metabolism of microorganisms and to perform cell design. For example, FBA was used to study growth capability under different carbon, nitrogen and phosphorous sources of the strain *Salinispora tropica* using a GSM (iCC908) (Contador et al., 2015). Another example is the study of the electron transfer during chemolithoautotrophic growth by *Acidithiobacillus ferrooxidans* (Campodonico et al., 2016).

The algorithm, Flux Scanning based on Enforced Objective Flux (FSEOF), developed by (Choi et al., 2010) is used to study the metabolic pathways that have an increased flux while enforcing a specialised metabolite production. FSEOF has been successfully implemented to find gene targets for overexpression, for example, increase actinorhodin in *S. coelicolor* by 52 fold (Kim et al., 2014) and to increase lycopene production in *E. coli* by 8 fold (using the result of FSEOF and MOMA to predict knockouts) (Choi et al., 2010).

A GSM can be constructed and easily subjected to FBA or MOMA in python programming language version 2.7 (https://www.python.org/download/releases/2.7/) using the COnstrain Based Reconstruction Analysis for python (COBRApy) toolbox (Ebrahim et al., 2013). Implementation of other techniques like FSEOF can also be accomplished using COBRApy in python.

Description of the thesis

The aim of this thesis was to explore the potential of *S. leeuwenhoekii* C34 to produce specialised metabolites by studying the metabolism through a GSM and identification of BGCs by genome mining. The work developed in this thesis will be presented in two chapters. In the first chapter, it is presented a high quality genome sequence of *S. leeuwenhoekii* C34 obtained by the combination of Illumina Miseq and PAC-Bio sequencing techniques. This reliable genome sequence was used to perform genome mining and identify BGCs of specialised metabolites, with special focus on identification of putative novel PKSs and NRPSs. Experimental work was done to try to identify the product of a BGC encoding for a hybrid *trans*-AT PKS/NRPS. In the second chapter, it is be described a user-friendly interface for the construction of GSMs and how it was used to develop the GSM of *S. leeuwenhoekii* C34. The GSM was used to study the metabolism of *S. leeuwenhoekii* C34 and the precursor supply for specialised metabolite biosynthesis. It was also used to predict

gene deletion or overexpression that would enhance specialised metabolite production.

Objectives

General objective:

The general objective of this thesis was to study the metabolism of *S. leeuwenhoekii* C34 with emphasis in specialised metabolite production pathways in order to find gene targets for metabolic engineering that would enhance their production.

The specific objectives were:

1. Identification of biosynthesis gene clusters of specialised metabolites in *S. leeuwenhoekii* C34 by genome mining.

2. To study the product of a novel biosynthetic gene cluster *in vivo*.

3. To develop a genome scale model of *S. leeuwenhoekii* C34 to analyse its metabolism.

4. To identify metabolic engineering targets that would enhance the production of a specialised metabolite, like chaxamycins, in *S. leeuwenhoekii* C34.

Chapter 1

Identification and characterization of gene clusters of *Streptomyces leeuwenhoekii* C34

1.1 Abstract

S. leeuwenhoekii C34 is a novel strain isolated from the Chaxa lagoon of the Atacama Desert, Chile. It posses antibiotic activity against *Staph. aureus* MRSA due the action of the specialised metabolites: chaxamycins and chaxalactins. A reliable genome sequence of *S. leeuwenhoekii* C34 was obtained using Illumina Miseq and PAC-bio RS II SMRT technologies. Thirthy-four biosynthetic gene clusters (BGCs) were found in the genome of *S. leeuwenhoekii* C34, plus one BGC encoded in pSLE2. Three lasso-peptide BGCs designated as lasso-peptide 1, 2 and 3, were identified. Lasso-peptide 3 was identified in the supernatant of *S. leeuwenhoekii* C34 grown in TSB/YEME culture media and it was also expressed in the heterologous host *S. coelicolor* M1152. It was confirmed that this lasso-peptide was the same that the chaxapeptin described for *S. leeuwenhoekii* C58. Also, a 64 kb BGC was identified encoding for a hybrid *trans*-AT PKS/NRPS that spans from 1083651 to 1147687 nt, that probably produces a halogenated specialised metabolite due to the presence of the gene *sle09470* that encodes for a chlorinating enzyme. In order to study the hybrid *trans*-AT PKS/NRPS BGC, nine strains derived from *S. leeuwenhoekii* C34 that could have an effect on the generation of the product of BGC were developed. The BGC was also cloned in heterologous hosts *S. coelicolor* M1152, M1154 and *S. albus*. Through HPLC MS/MS analysis and metabolite profile comparison a group of compounds was identified with a chlorinated pattern, however they were not exclusive to *S. leeuwenhoekii* strains because they were also observed in the supernatant of *S. coelicolor* M1152, so they were rejected as possible products of the hybrid *trans*-AT PKS/NRPS. On the other hand, a metabolite *m/z* 611.53 $[M + H]^+$ was detected only in the strains *S. leeuwenhoekii* M1614 (Δchaxamycin BGC) and M1619 (Δchaxamycin BGC; Δ*sle09560*). Further studies are needed in order to confirm if the differently expressed metabolite correspond to a product of the hybrid *trans*-AT PKS/NRPS BGC.

1.2 Introduction

The genes encoding for specialised metabolites are usually clustered in biosynthetic gene clusters (BGCs). Usually, the BGCs are composed not only by biosynthetic genes, but also with post-synthesis modification genes, regulators, transporters, resistance genes. This type of clustered configuration facilitates the identification of putative BGCs through genome mining, because with the identification of a single gene it is possible to discover the whole biosynthesis pathway (net). Also, availability of genome sequences has allowed the identification of BGC encoding for specialised metabolites that does not have an associated product (silent or cryptic BGC), that potentially encode for novel specialised metabolites (Ōmura et al., 2001; Bentley et al., 2002; Ikeda et al., 2003). For example, through genome mining it has been possible to discover the non-ribosomal peptide coelichelin in *S. coelicolor* (Lautru et al., 2005), the macrolide stambomycins A–D in *S. amb-ofaciens* (Laureti et al., 2011) and a hybrid PKS-NRPS in *Aspergillus nidulans* (Bergmann et al., 2007).

A type of specialised metabolites are the lasso-peptides. They are ribosomally-synthesised and subjected to post-synthesis modifications to produce a characteristic structure that includes a ring, a loop and a tail (Maksimov et al., 2012; Pan et al., 2012; Elsayed et al., 2015). This structure gives them high thermal and chemical stability (Zimmermann et al., 2013). The proteins needed for their biosynthesis and processing are clustered together. The precursor peptide is divided in leader and core (or lasso) peptide, the penultimate amino acid of the precursor peptide is a conserved threonine, and the core peptide begins with a glycine residue. The ring is formed by a covalent bond between the glycine and the side chain of either a glutamate or aspartate residue located at 6-10 residues of distance (Maksimov et al., 2012). They could have broad bioactivities such as antimicrobial, antiviral or anticancer (Um et al., 2013; Elsayed et al., 2015). For example, the lasso-peptide MccJ25 have antimicrobial activity due to inhibition of RNA polymerase. This inhibition is produced due to the structure of the peptide rather than the sequence (Maksimov et al., 2012).

Other types of specialised metabolites are polyketides and non-ribosomal peptides, synthesised by PKS and NRPS, respectively. Among these BGCs it is possible to find halogenating enzymes, that could modify the molecule synthesised by the main biosynthetic genes. Naturally occurring halogenated compounds are of great interest in the pharmaceutical industry because they usually show enhanced bioactivities when compared with the non-halogenated version of the compound (Gribble, 2004; Vaillancourt et al., 2005). For example, the bioactivity of the antibiotic vancomycin is reduced by 30 % and 50 % when one or two of the chlorine atoms is removed, respectively (Harris et al., 1985).

In this work, the *S. leeuwenhoekii* C34 genome was sequenced *de novo* and it was used to identify BGC encoding for specialised metabolites, with special focus in BGC of lasso-peptides, polyketides, non-ribosomal peptides, and possible halogenated compounds. Metabolite profile comparison, through HPLC MS/MS analysis, was used to detect the product of BGCs.

1.3 Methodology

1.3.1 Bacterial strains and general procedures

A list of the strains used in this work is available in Appendix A Table A.1. *S. leeuwenhoekii* C34 was isolated from the Atacama Desert by a collaboration with Prof. Alan Bull (University of Kent) and Prof. Michael Goodfellow (University of Newcastle). All the strains were kept in glycerol 20% at -4 and -80 °C. The experimental protocols used in this work are described in the Appendix D.

1.3.2 Sequencing of *S. leeuwenhoekii* C34 genome

The genome of *S. leeuwenhoekii* C34 was sequenced *de novo* using combined NGS technologies (Gomez-Escribano et al., 2015). Briefly, the genomic DNA of *S. leeuwenhoekii* C34 was isolated using a standard methodology (Kieser et al., 2000) and sent for sequencing with Illumina Miseq (Department of Biochemistry, University of Cambridge, Cambridge, UK) technology and PAC-bio RS II SMRT (The Earlham institute, formerly known as the sequencing technique Genome Analysis Centre (TGAC), Norwich Research Park, Norwich, UK). The PAC-bio sequencing results was used as reference and was curated with the Illumina Miseq sequence data to obtain a high quality genome sequence. The annotation of the genome was done using Prodigal (Hyatt et al., 2010), followed by BASys (Van Domselaar et al., 2005) for assignment of putative functions, furthermore RAST (Aziz et al., 2008; Overbeek et al., 2014) was used to annotated tRNA and rRNA genes.

1.3.3 Identification of putative BGCs for specialised metabolites biosynthesis

The identification of putative BGCs encoding for PKSs, NRPSs, siderophores, terpenes, among others, was done using AntiSMASH (Medema et al., 2011a; Blin et al., 2013; Weber et al., 2015). Briefly, AntiSMASH uses libraries of profile hidden markov models (pHMM) to define and to analyse BGCs, later the identified BGCs are compared to known clusters. The domains of the biosynthethic genes of specialised metabolites are identified using other library of pHMMs. The output is visualised in an interactive website (Medema et al., 2011a). Then, each putative BGC identified by antiSMASH was evaluated for consistency and characterization. With special focus in the identification of boundaries of PKSs and NRPSs, by comparison of the putative BGC with the most similar known BGC and/or the identification of conserved gene regions. Lasso-peptides BGCs were not identified with version 1 or 2 of AntiSmash, so they were identified manually by comparison with the lariatin BGC (Inokoshi et al., 2012). With AntiSmash version 3, it was possible to directly identify the lasso-peptides. Domain analysis of the putative PKSs, NRPSs and lasso-peptides was done using the conserved domain database (CDD) (Marchler-Bauer and Bryant, 2004; Marchler-Bauer et al., 2009, 2011, 2014).

1.3.4 Generation of mutant strains to study BGCs

Different mutant strains were developed in order to study the product of their BGC (see strains developed in this work in Table A.1). Generation of gene knockouts were accomplished by cloning the flanking regions of the gene of interest in the temperature sensitive plasmid pGM1190, followed by cloning a selection marker in between the previously cloned regions. Over-expression of genes was accomplished by cloning the gene of interest downstream a strong constitutive promoter (*erm*E*) in the plasmid pIJ10257. These plasmids are conjugated to *S. leeuwenhoekii* C34 using the protocol described in Appendix D.6.

1.3.5 Metabolite profile comparison analysis

Comparison of the metabolite's profile of the different strains generated in this work, was done by HPLC MS/MS analysis using either of the following instruments: a Shimadzu LC-MS system coupled to LCMS-IT-ToF mass spectrometer, a Bruker Daltonik electrospray-ion tramp ESI-IT Esquire 4000 or Maxis II QTOF. For details of the chromatographic conditions used for each instrument, see Appendix D Section D.10.

1.3.6 Implementation and software usage

HPLC-MS/MS results analysis was done using either Mass++ (http://www.masspp.jp/) or Mzmine 2 (Pluskal et al., 2010), installed in a computer with 64-bit Windows 10 system, Intel® Core^{TM} i5-2430M CPU @ 2.40 GHz with 8 GB RAM. Draws of BGC were done using a home made python script (Appendix E.1.1), and edited using Inkscape (https://inkscape.org).

1.4 Results

From the sequencing with Illumina Miseq 279 contigs assembled into 175 scaffolds were obtained. While with the PacBio sequencing platform one large contig (containing all the chromosome) and two small contigs (corresponding to plasmids) were obtained. A summary of the genomic information is shown in Table 1.1. The details of the curation and generation of the genome sequence are available at (Gomez-Escribano et al., 2015).

1.4.1 BGC in the genome of *S. leeuwenhoekii* C34

The Antismash version 3.0.4 output gives a total of 34 gene clusters in the genome of *S. leeuwenhoekii* C34, and one gene cluster in pSLE2, for a total of 35 gene clusters (Table 1.2). The previous version of Antismash was not able to identify the gene clusters encoding for lasso-peptides (Gomez-Escribano et al., 2015). A great variety of BGCs (Figure 1.1) were found, and almost half of them (47.2 %) are either of PKS type, NRPS or hybrids.

Table 1.1: Characteristics of the *S. leeuwenhoekii* C34 genome. Adapted from (Gomez-Escribano et al., 2015).

Assembled chromosome size	7903895 bp
Estimated chromosome size	8285171 bp
Estimated terminal inverted repeats	388272 bp
Chromosome topology	Linear
Chromosome G + C content	73%
rRNA operons	6
tRNA genes	65
pSLE1 circular plasmid	86370 bp
pSLE1 G + C content	69%
pSLE2 linear plasmid	132226 bp
pSLE2 G + C content	70%
Putative BGCs for specialised metabolites	34 (+1 in pSLE2)

As shown in Table 1.2 the genomic potential of *S. leeuwenhoekii* C34 for producing specialised metabolites is great and only a few of them have been detected in production media (Rateb et al., 2011a,b). From the total of specialised metabolites BGC found in *S. leeuwenhoekii* C34, only 22 % are associated to known compounds. Among them there are the BGCs for hygromycin, desferrioxamines, chaxamycins and chaxalactins.

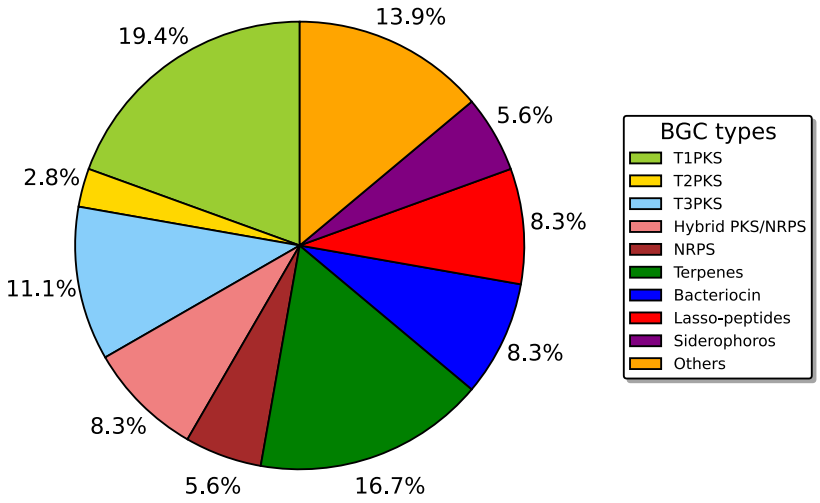


Figure 1.1: Percentages of the different BGC types found in *S. leeuwenhoekii* C34 genome using Antismash version 3.0.4. For more information see Table 1.2

The remaining BGCs of PKS and NRPS encoding for biosynthesis pathways of unknown compounds, were subjected to further study in order to confirm if they were unbroken and to define their boundaries. To do so, the genome context of the BGC was analysed, and also when possible, it was compared to similar BGCs. For several of the BGCs the assignation of boundaries was very difficult since the regions surrounding the BGC did not have homologous genes to other strains, and the BGC did not have a known similar BGC.

Table 1.2: Gene clusters of *S. leeuwenhoekii* C34.

No.	Type[1]	From	To	Curated annotation
1	T1pks	*115304*	*131991*	This work
	Not identified	*160425*	*189028*	Hygromycin A, by (Gomez-Escribano et al., 2015) based on (Palaniappan et al., 2006; Rateb et al., 2011b)
2	T1pks	191701	240196	
3	T1pks-Nrps	324784	392261	
4	Nrps	*397624*	*411018*	This work
5	T3pks	416888	458084	
6	Bacteriocin	572464	582679	
7	Terpene	598795	619823	
8	Lassopeptide	*648373*	*654830*	Lasso-peptide 2, by (Gomez-Escribano et al., 2015)
9	Nrps	*734934*	*777465*	This work
10	Terpene	1056004	1076960	
11	Otherks-Transatpks-Nrps	*1083651*	*1147687*	Hybrid *trans*-AT PKS/NRPS, this work based on (Cheng et al., 2002; Tang et al., 2004)
12	T1pks-Terpene	*1210347*	*1290550*	Chaxamycins, by (Castro, 2015)
13	T1pks	1497127	1544539	
14	Terpene	1624097	1645110	
15	T1pks-Siderophore	1776281	1833813	
16	Terpene	1972277	1994487	
17	Bacteriocin	2013690	2025087	
18	Siderophore	2293580	2305424	
19	Nrps-T1pks	2668194	2719415	
20	T3pks	2937137	2978264	
21	Terpene	*3056325*	*3058819*	Albaflavenone, by (Gomez-Escribano et al., 2015) based on (Lin et al., 2006; Zhao et al., 2008)
22	Lassopeptide	*3560004*	*3565538*	Lasso-peptide 1 (This work)
23	Bacteriocin	3683199	3693399	
24	Siderophore	*5237176*	*5244356*	Desferrioxamine E, by (Gomez-Escribano et al., 2015) based on (Barona-Gómez et al., 2004)
25	Melanin	5330379	5340933	Melanin
26	Aminoglycoside-Butyrolactone	5385171	5417416	
27	Ectoine	*6180396*	*6183663*	Ectoine, this work based on (Peters et al., 1990; Bursy et al., 2008)
28	Other	6710095	6751819	
29	T3pks	6822979	6864043	
30	T1pks	*7146903*	*7227608*	Chaxalactins, by (Castro, 2015)
31	T1pks	7355977	7439461	
32	Other	7486047	7529121	

Table 1.2: Gene clusters of *S. leeuwenhoekii* C34 (continued).

No.	Type[1]	From	To	Curated annotation
33	Terpene-T2pks	7530162	7588405	
34	Terpene	7744176	7768730	
35	Lassopeptide	*pSLE2(103389)*	*pSLE2(105999)*	Lasso-peptide 3, by (Gomez-Escribano et al., 2015)

[1]According to AntiSMASH. In italics are shown the boundaries defined previously or in this work (see column Curated annotation).

Among the list of specialised metabolite BGCs, there are a few that were further studied: Lasso-peptides 1, 2 and 3 (see Section 1.4.2) and the hybrid *trans*-AT PKS/NRPS (see Section 1.4.3).

1.4.2 Lasso-peptides

Lasso-peptides are ribosomal synthesised and post-translationally modified peptides, they have a characteristic structural conformation that confer them with diverse bioactivities and are synthesised in small BGCs (Maksimov et al., 2012; Pan et al., 2012; Elsayed et al., 2015). As shown in Table 1.2 there are three BGCs encoding for lasso-peptides. Each BGC has at least four genes encoding for a precursor peptide, lasso-peptide cyclase, protein of unknown function, and lasso-peptide protease. Additionally, lasso-peptides 1 and 2 have genes encoding for transporters (Figure 1.2 and Table 1.3 and 1.4).

Table 1.3: Description of the proteins found in the lasso-peptide 1 BGC and closest NCBI database homologous.

Gene	Length (aa)	Proposed function or domain	Protein homolog (NCBI accession number) (similarity/identity %)
sle29710	647	ABC transporter	ABC transporter *Streptomyces pactum* (WP_055417674) (84/79)
sle29720	144	Transglutaminase-like superfamily	polyketide beta-ketoacyl synthase *Streptomyces cyaneogriseus* (WP_044383228) (99/99addlabel)
sle29730	84	Coenzyme PQQ synthesis protein D	Coenzyme PQQ synthesis protein D (PqqD) *Streptomyces* sp. SceaMP-e96 (SCK51853) (75/64)
sle29740	618	Asparagine synthase	asparagine synthase *Streptomyces cyaneogriseus* (WP_044383227) (98/97)
sle29750	36	Precursor peptide	hypothetical protein SGLAU_31040 *Streptomyces glaucescens* (AIS02142) (97/94)

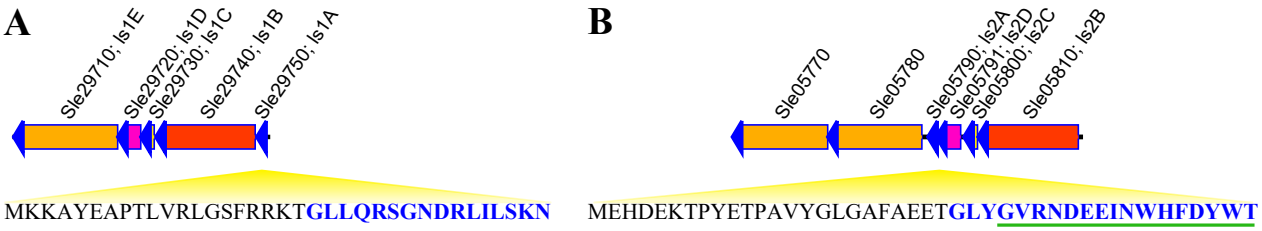


Figure 1.2: Lasso-peptide 1 (**A**) and lasso-peptide 2 (**B**) BGC. Highlighthed is the amino acids precursor sequence encoded by the *sle29750* and *sle05790* for each BGC, respectively. In blue, is shown the amino acid sequence that should correspond to the lasso-peptides. In underlined green is shown an alternative possible sequence of the lasso-peptide 2. For more information see Table 1.3 and Table 1.4.

Table 1.4: Description of the proteins found in the lasso-peptide 2 BGC and closest NCBI database homologous.

Gene	**Length (aa)**	**Proposed function or domain**	**Protein homolog (NCBI accession number) (similarity/identity %)**
sle05770	593	ABC transporter	ATP-binding cassette, subfamily C *Micromonospora echinofusca* (SCG15655) (58/45)
sle05780	586	ABC transporter	ABC-transporter *Actinomadura namibiensis* (CAX48974) (59/48)
sle05790	43	Precursor peptide	hypothetical protein GA0070563_12268 *Micromonospora carbonacea* (SCF49478) (68/58)
sle05791	151	Transglutaminase-like superfamily	polyketide beta-ketoacyl synthase *Streptomyces griseochromogenes* (WP_067302413) (67/59)
sle05800	85	Coenzyme PQQ synthesis protein D	hypothetical protein *Streptomyces griseochromogenes* (WP_067302410) (61/48)
sle05810	625	Asparagine synthase	asparagine synthase (glutamine-hydrolysing) *Micromonospora carbonacea* (SCF49484) (62/49)

The predicted amino acid sequence of the lasso-peptide 1 is also found in the hypothetical protein TUE45_07344 (CUW32593) of *S. reticuli*, however it has not been described any lasso-peptide for that species. On the other hand, the predicted amino acid sequence of the lasso-peptide 2 is not found in any other described species. The predicted amino acid sequence of the lasso-peptide 3 was the same than the one described for chaxapeptin, compound that is synthesised by *S. leeuwenhoekii* strain C58 (Elsayed et al., 2015), a close relative of *S. leeuwenhoekii* C34. The precursor proteins of lasso-peptides 1 and 2 have the conserved threonine in the last position of the leader sequence, while the precursor of lasso-peptide 3 has it in the penultimate position.

The protease enzyme is proposed to be in charge of cutting the precursor peptide to obtain the core peptide. A catalytic triad of cysteine-histidine-aspartate present in homologous proteins of Sle29720, Sle05791 and Sle2_133, such as for McjB (Accession No. Q9X2V8) of the microcin J25 lasso-peptide (Pan et al., 2012) has been described. However, in the *S. leeuwenhoekii* C34 proteins mentioned, the triad cysteine-histidine-glutamate has been found. The aspartate and glutamate amino acids have similar physicochemical properties, and this change of amino acids has been found in the functional protein LarD from the lariatin BGC (Figure 1.3). Also, the mutation of the aspartate residue to alanine in the McjB protein has been studied and it has been found that the production of the final lasso-peptide is possible, although it is diminished, while mutation of either the cysteine or histidine to alanine are proved to be deleterious for the production of microcin J25 (Pan et al., 2012). So, the change from aspartate to glutamate residue should not produce the inactivation of the enzyme.

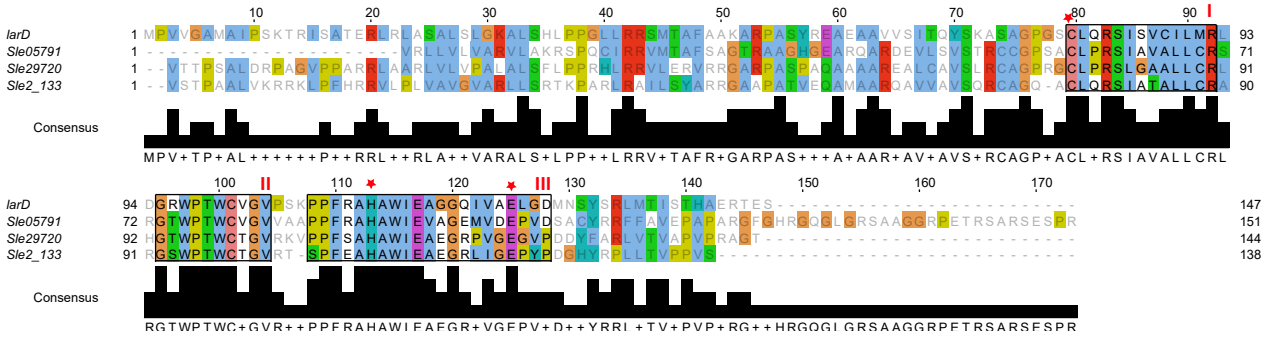


Figure 1.3: Alignment of the amino-acids of the proteins Sle29720, Sle05791 and Sle2_133, and their homologous protein of lariatin BGC. The star indicates the residues of the catalytic triad. LarD (NCBI accession number: BAL72549)

The protein in charge of the cyclization of the lasso-peptides 1, 2 and 3 are, Sle29740, Sle05810 and Sle2_131, respectively. These proteins are homologous to LarB from the lariatin BGC. All of them have a conserved ATP binding pocket (see supplementary Figure E.2).

Despite the fact that the lasso-peptide 3 BGC, has 4 genes (Figure 1.4 and Table 1.5) and that a transporter is missing, the lasso-peptide 3 was detected in the supernatant and mycelium extract of *S. leeuwenhoekii* C34 grown in TSB/YEME, and it was also successfully expressed in *S. coelicolor* M1152 heterologous host (data not shown). The export of lasso-peptide 3 could be due to non-specific flux pumps. Additional analyses confirmed that the lasso-peptide 3 had the same structure than the chaxapeptin described in *S. leeuwenhoekii* strain C58 (data not shown).

Figure 1.4: Lasso-peptide 3 BGC. The amino acids precursor sequence encoded by the *Sle2_130* gene is highlighted. In blue is shown the amino acid sequence of the lasso-peptide. For more information see Table 1.5.

Table 1.5: Description of the proteins found in the lasso-peptide 3 BGC and closest NCBI database homologous.

Gene	Length (aa)	Proposed function or domain	Protein homolog (NCBI accession number) (similarity/identity %)
sle2_130	44	Precursor peptide	hypothetical protein SBI_02670 *Streptomyces bingchenggensis* BCW-1 (ADI05791) (76/63)
sle2_131	604	Asparagine synthase	asparagine synthetase *Streptomyces kanamyceticus* (WP_055550888) (73/59)
sle2_132	85	Coenzyme PQQ synthesis protein D	hypothetical protein *Streptomyces* sp. CdTB01 (WP_058924575) (72/54)
sle2_133	138	Transglutaminase-like superfamily	Transglutaminase-like superfamily protein *Streptomyces* sp. SceaMP-e96 (SCK29524) (85/69)

1.4.3 Hybrid *trans*-AT PKS/NRPS BGC

A BGC that spans a region of 64 kb encoding for a hybrid *trans*-AT PKS/NRPS BGC has been identified in the *S. leeuwenhoekii* C34 genome from position 1083651 to 1147687 nt. This BGC encodes for a putative novel specialised metabolite, since it has not been described previously. The main biosynthetic genes are *sle09500*, *sle09480*, *sle09450* and *sle09440*. There are also other genes associated with post-synthesis modification, transporters and regulators (Figure 1.5).

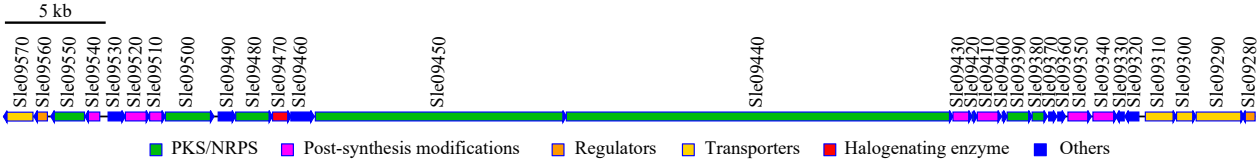


Figure 1.5: Hybrid *trans*-AT PKS/NRPS BGC. For more information see Table E.1.

The most similar known BGC is the leinamycin BGC. Leinamycins are specialised metabolites synthesised by *S. atroolivaceus* and they have antibiotic and antitumor activity (Hara et al., 1990; Gates, 2000; Cheng et al., 2002; Tang et al., 2004). The leinamycin BGC includes 28 genes (from lnmA to lnmZ'), all of them but three (lnmB, lnmC and lnmS) have homologous genes in the gene cluster of the hybrid *trans*-AT PKS/NRPS. On the other hand the hybrid *trans*-AT PKS/NRPS has five genes, *sle09560* (*tet*R transcriptional regulator), *sle09540* (methyltransferase), *sle09470* (chlorinating enzyme), *sle09390* (Beta-ketoacyl-acyl synthase) and *sle09330* (Glyoxalase/Bleomycin resistance protein/dioxygenase), that does not have homologous genes in the leinamycin gene cluster (Figure 1.6, Table E.1). The similarities of this two gene clusters will help in the prediction of the possible compound of the hybrid *trans*-AT PKS/NRPS BGC.

The hybrid *trans*-AT PKS/NRPS BGC has two NRPS modules and six PKS modules plus a gene specifically encoding for the AT domain (see top of Figure 1.8). The characteristics of each domain are:

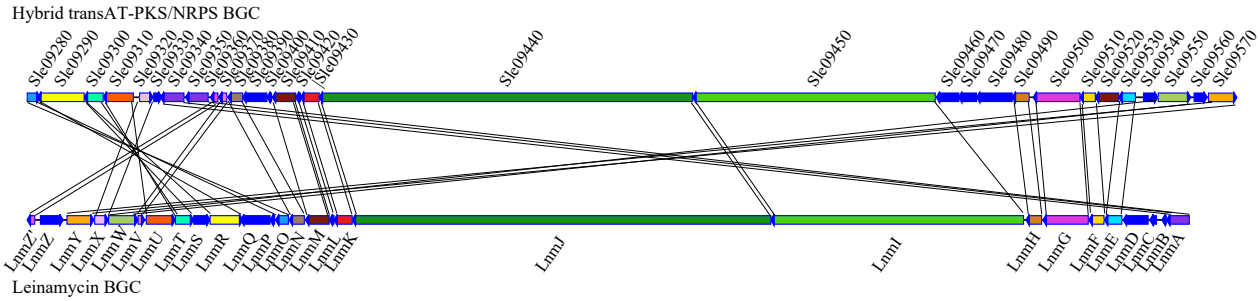


Figure 1.6: Comparison of the hybrid *trans*-AT PKS/NRPS and leinamycin BGC. Same colours indicate homologous genes. For more information see Table E.1

AT domain. The AT domain is encoded in an independent gene, named *sle09500*. The domain has the GHSxG and the APFHS signature. It is predicted that the product of this gene will recognize malonyl-CoA units.

A domain. There are two A domains in the hybrid *trans*-AT PKS/NRPS BGC. The first is located in *sle09480* and is predicted to use the amino acid threonine, while the second is located in *sle09450* and is predicted to incorporate a cysteine residue to the non-ribosomal peptide chain.

KS domains. The gene cluster contains eight KS domains. The fingerprint TxCxxS and the histidine residues of HGTGT and GSVKxxxGH motifs are present in six KS domains. KS1 lacks the catalytic histidine of HGTGT motif (Figure E.3), so is assigned as $KS1^0$ and it would not elongate the polyketide chain (Helfrich and Piel, 2016). KS8 lacks the catalytic cysteine and is synthesised in a gene outside the main biosynthetic genes; it is probable that KS8 is involved in β-branching.

KR domains. All the KR domains have the catalytic triad lysine-serine-tyrosine. The Rossmann fold motif, GxGxxGxxxA, required for NADP(H) binding is present in all the KR but in three of them it has small variations (Figure E.4).

ACP and PCP domains. All the ACP and PCP domains have the fingerprint GxxS (Figure E.5 and E.6). There are more than one ACP in module 3 and 8, having more that one ACP domain within a module is a common characteristic of *trans*-AT PKS. Studies have demonstrated that in such cases a skipping mechanism is possible where either of the ACP domains could carry the polyketide chain (Tang et al., 2006). Also, extra ACP domains have been related to the introduction of β-branching into the molecule (Calderone, 2008; Piel, 2010).

DH domains. There is one DH domain and it has the **H**xxxGxxxxP and **D**LAAL motifs. This domain is predicted to be active since it has the catalytic dyad of histidine and aspartate.

Based on the gene sequences there are two possible regions subjected to β-branching. The first one is proposed to happen in module 5, and the second one is proposed to happen in module 8 and be similar to the one described for leinamycin BGC.

There are four genes, *lnmFKLM*, involved in β-branching in the leinmyacin BGC. Those genes

have homologous genes in the hybrid *trans*-AT PKS/NRPSBGC, *sle09510* and *sle09430-10* (Table E.1). LnmK has a bifunctional activity of AT/decarboxylase (DC) (Liu et al., 2009; Lohman et al., 2013), the acyltransferase activity of this gene is related to a tyrosine residue instead of serine (like in traditional AT domains). The homologous protein in *S. leeuwenhoekii* C34, Sle09430, has the conserved tyrosine necessary for the activity of the enzyme.

Other features of the BGC of the hybrid *trans*-AT PKS/NRPS

Within this BGC there is a gene, *sle09470*, encoding for an enzyme that could be responsible for chlorinating the product of the hybrid *trans*-AT PKS/NRPS, hybrid PK-NP. Comparison of the amino-acid sequence of the protein Sle09470 with related proteins associated with chlorination of compounds is shown in Figure 1.7, percentages of identities and coverage are over 45 % and 84 %, respectively. CmaB a member of the non-haem Fe^{2+}, α-ketoglutarate dependent enzyme superfamily, was the first of its class described to have chlorinating activity (Vaillancourt et al., 2005).

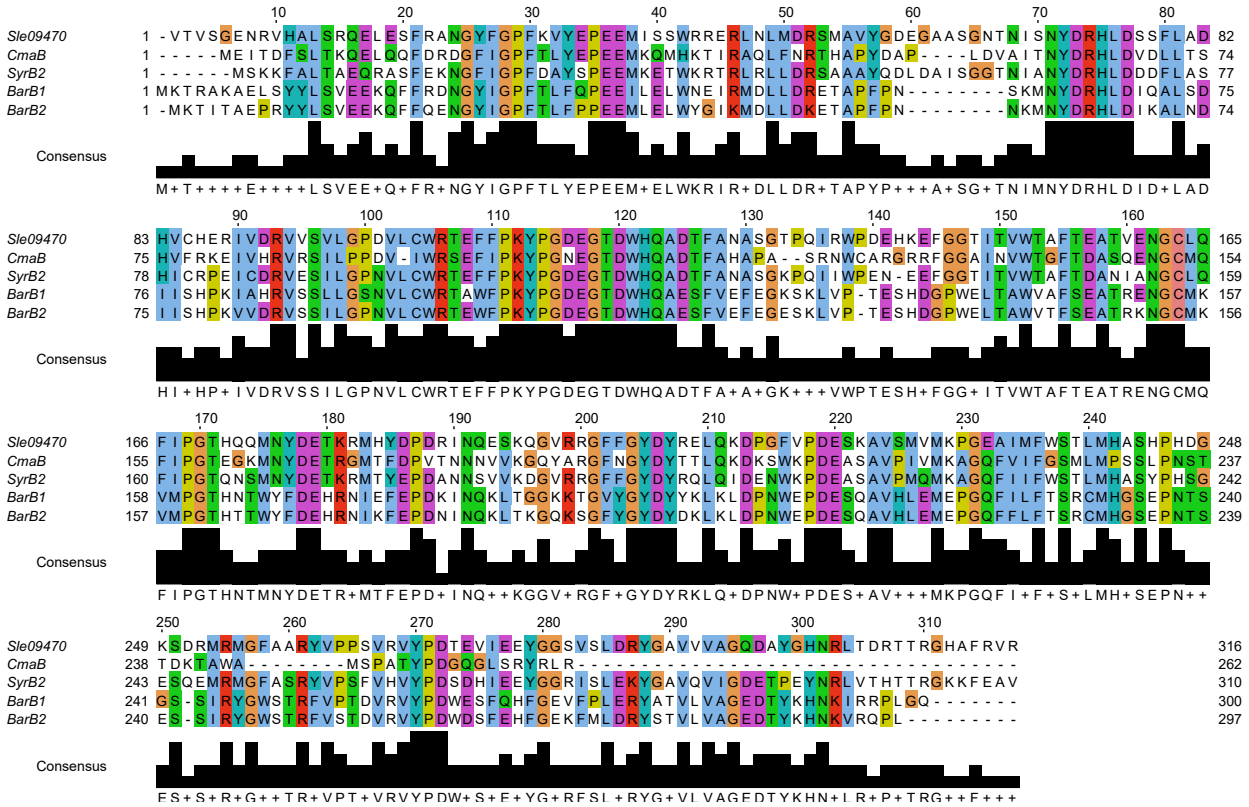


Figure 1.7: Alignment of the amino-acids of the putative chlorinating enzyme (Sle09470) detected in the hybrid *trans*-AT PKS/NRPS biosynthetic gene cluster, with similar known chlorinating enzymes. CmaB (NCBI accession number: AAC46036); SyrB2 (NCBI accession number: AAD50521); Barb1 (NCBI accession number: AAN32975); Barb2 (NCBI accession number: AAN32976).

Within the defined boundaries of the hybrid *trans*-AT PKS/NRPS BGC there are two regulatory proteins, encoded by *sle09560* and *sle09280*. Both of them could be associated to the expression of the BGC. Sle09560 has a TetR domain and it has a 47 % of identity with 86 % of coverage with TetR family transcriptional regulator from *S. tsukubensis* (NCBI accession number: WP_040914130).

Alignment of these proteins is shown in Figure E.8.

Sle09280 shares an 85 % of coverage with 49 % of homology to the only regulator that is present in the leinamycin BGC, LnmO (NCBI accession number: AAN85528). It also shares 95 % coverage with 47 % of homology to the Crp/Fnr family transcriptional regulator from *Catenulispora acidiphila* (NCBI accession number: WP_012786802). Alignment of the proteins is shown in Figure E.7. LnmO has been proposed to be an activator (Tang et al., 2006) therefore Sle09280 was further analysed as an activator.

Outside of the proposed boundaries of the BGC there is a gen, *sle09760*, that has domains associated with the LysR transcriptional regulator and is highly similar to the LysR family transcriptional regulator from *S. cyaneogriseus* (NCBI accession number: WP_044386206) with 99 % of identity and 100 % coverage. This gene was also taken in consideration when studying the BGC.

1.4.4 Prediction of the possible structure of the hybrid PK-NP

The structure of compounds synthesised by PKS type I can be easily inferred from the DNA sequence. However for *trans*-AT PKS it is not easy to predict their product since they usually have other conformations (Piel, 2010; Helfrich and Piel, 2016). Nevertheless, attempts to infer the possible structure of the core of the compound were done. For doing this all the information of fingerprints of each domains were taken into account, and the similarities of the hybrid *trans*-AT PKS/NRPS BGC to the leinamycin BGC. The proposed biosynthetic pathway is shown in Figure 1.8.

1.4.5 Generation of mutants to study the production of the hybrid PK-NP

Several knockouts genes had been done in order to carry out the identification of the hybrid PK-NP by metabolic profiling.

***Trans*-AT knockout.** The *trans*-AT gene encoded by *sle09500*, is the gene in charge of the recognition of the units that would be incorporated to the polyketide chain. Knockout of this gene would avoid the production of the product of the gene cluster. Flanking regions of the *trans*-AT gene were cloned in a temperature sensitive vector pGM1190 yielding pIJ12805. Later, the kanamycin resistance gene (neo^R) was cloned in the middle of the flanking regions to yield pIJ12803. *E. coli* ET12567/pUZ8002 was transformed with pIJ12803, then the *E. coli* strain was used as donor to transfer the plasmid through conjugation to *S. leeuwenhoekii* C34 and strain *S. leeuwenhoekii* M1601 that does not contain the *trans*-AT gene was obtained.

Halogenase knockout. As stated previously the gene cluster for the hybrid *trans*-AT PKS/NRPS has a gene named *sle09470* encoding for a putative halogenase. It has been detected using inductively coupled plasma (ICP), that this strain produces a chlorinated compound when using a DM supplemented with chlorine (Prof. M. Jaspars, personal communication). Similarly to the previous case, flanking regions of *sle09470* were cloned in pGM1190 to yield pIJ12804. Then, neo^R

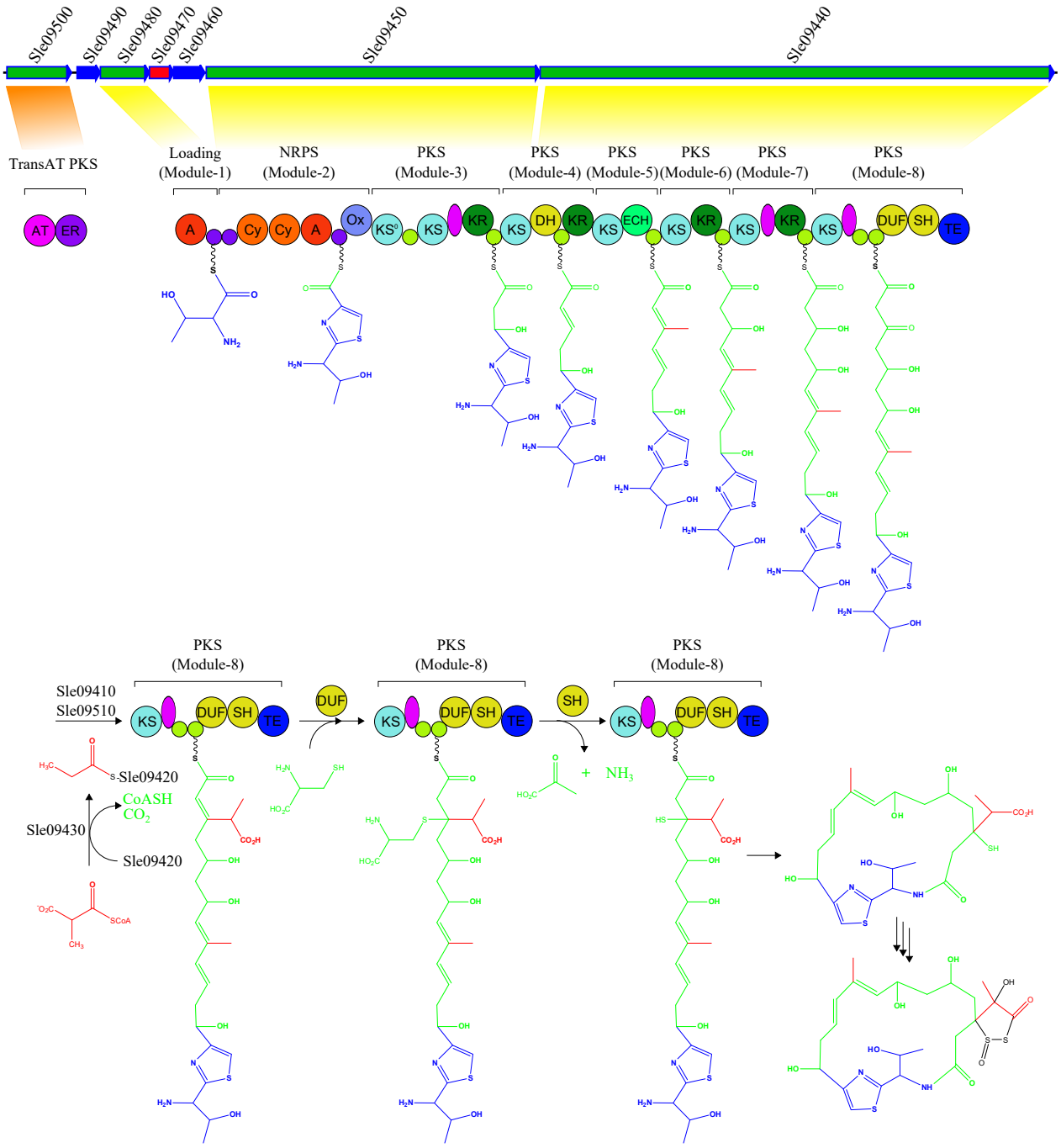


Figure 1.8: Prediction of the possible structure of the product of the hybrid *trans*-AT PKS/NRPS biosynthetic gene cluster, hybrid PK-NP. The biosynthesis pathway is based in the biosynthesis of leinamycin (Ma et al., 2015). Blue, building blocks incorporated by NRPS; Green, building blocks incorporated by PKS; Red, β-branching; Black, other modifications. The purple circle denote PCP (peptidyl-carrier protein) domains, and the light green represent ACP (acyl-carrier protein) domains. A, adenylation domain; AT, acyltransferase; Cy, condensation/cyclization; DH, dehydratase; DUF, domain of unknown function; KR, ketoreductase; KS, ketosynthase; MT, methyltransferase; Ox, oxidation; SH, PLP-dependent cysteine lyase domain; TE, thioesterase.

gene was ligated between the flanking regions to achieve pIJ12801. The plasmid was transferred to *S. leeuwenhoekii* C34 through conjugation and later selection of double recombinants allowed to obtain the strain *S. leeuwenhoekii* M1600.

Deletion of the chaxamycin biosynthetic gene cluster. Chaxamycins are one of the main specialised metabolites produced by *S. leeuwenhoekii* C34 (Rateb et al., 2011b). As in the case of the hybrid PK-NP, chaxamycins also use malonyl-CoA as building blocks (among others). Deletion of this BGC would increase the availability of precursors for the biosynthesis of other specialised metabolites.

To accomplish the deletion of the chaxamycin BGC, a region of 2.5 Kbp from each side of the chaxamycin BGC was cloned in pGM1190 (apramycin resistant *aac*(3)-IV) to yield pIJ12809. The plasmid pIJ12809 was transferred to *S. leeuwenhoekii* M1653 (ΔAHBA *neo*^{R}) by conjugation. Single crossovers ex-conjugants were selected with apramycin. Identification of double recombinants (*neo*^{S} and *aac*(3)-IV^S) was done through replica plates. The candidates for double recombinants were confirmed by PCR. *S. leeuwenhoekii* C34 Δchaxamycins was named M1614.

Deletion of *sle09560* (*tetR*). The deletion of the putative negative regulator *sle09560* was fulfilled by cloning flanking regions of the *sle09560* gene in pGM1190 to yield pIJ12814, then the *neo* gene was cloned between the two flanking regions to yield pIJ12813. The plasmid pIJ12813 was transferred to *E. coli* ET12567 pUZ8000 and conjugated to *S. leeuwenhoekii* C34 and *S. leeuwenhoekii* M1614 to yield M1615 and M1617 derivatives, respectively. Later, the *neo* gene was taken out of the genome of *S. leeuwenhoekii* M1617 by conjugation with pIJ12814 and selection of *neo*^{S} double recombinant clones. A double recombinant strain was obtained and confirmed by PCR, that they were sensitive to kanamycin; this strain was named M1619.

Overexpression of the regulatory genes: *sle09280* and *sle09760* (*lys*R). The overexpression of the regulators *sle09280* and *sle09760* was reached by cloning each gene under a strong promoter in the plasmid pIJ10257 yielding pIJ12817 and pIJ12810, respectively. The plasmid pIJ12817 was transferred into *E. coli* ET12567 pUZ8000 and conjugated to *S. leeuwenhoekii* C34 and *S. leeuwenhoekii* M1614 to yield M1616 and M1618, respectively. On the other hand pIJ12810 was conjugated into *S. leeuwenhoekii* C34, *S. coelicolor* M1607, *S. coelicolor* M1609, *S. albus* M1612 to yield M1605, M1608, M1611 and M1613, respectively.

Heterologous expression of the *trans*-AT PKS/NRPS BGC. A PAC library was constructed by BioS&T inc (Montreal, Canada). The library was scanned with the primers VR016 to VR019 (Table B.1) that amplified a ˜500 bp region at each side of the BGC of the hybrid *trans*-AT PKS/NRPS. Two clones designated as PAC 18O and PAC 14B were positives and used to transfer the BGC of the hybrid *trans*-AT PKS/NRPS through triparental mating to *S. coelicolor* M1152, *S. coelicolor* M1154 and *S. albus* J1074 (for information about the protocol used go to Section D.7).

To confirm that the PAC clones had the correct sequence, a PCR with internal primers was done (data not shown) and also the whole DNA was isolated and sent for sequencing. It was found that the PAC 18O contains two regions of *S. leeuwenhoekii* C34 genomic DNA, from 741835 to 873221 and from 1054450 to 1173025. The first segment does not include the hybrid *trans*-AT PKS/NRPS BGC but the second fragment does. The reason why the PAC clone has two separated segments of DNA is because during the generation of the PAC library the genomic DNA was digested with BamHI and then ligated to pESAC13, in the course of this process two segments digested with BamHI could be ligated. On the other hand, the PAC 14B contains the region from 1,070,660 to

1,287,284, that contain the whole hybrid *trans*-AT PKS/NRPS BGC plus almost all the chaxamycin BGC.

1.4.6 Metabolic profile comparison

Comparison of *S. leeuwenhoekii* C34, M1600 and M1601. Cultures grown in DM of *S. leeuwenhoekii* C34, M1600 (Δ*sle09470* halogenase gene knockout) and M1601 (Δ*sle09500 trans*-AT gene knockout) were carried out. It was expected to see the absence of a peak in the mutant strains compared to the WT, however such metabolite profile was not observed in neither positive nor negative ionization results (Figure 1.9). A higher intensity of the peaks corresponding to chaxamycins in the mutant strains was observed (see the arrow in Figure 1.9).

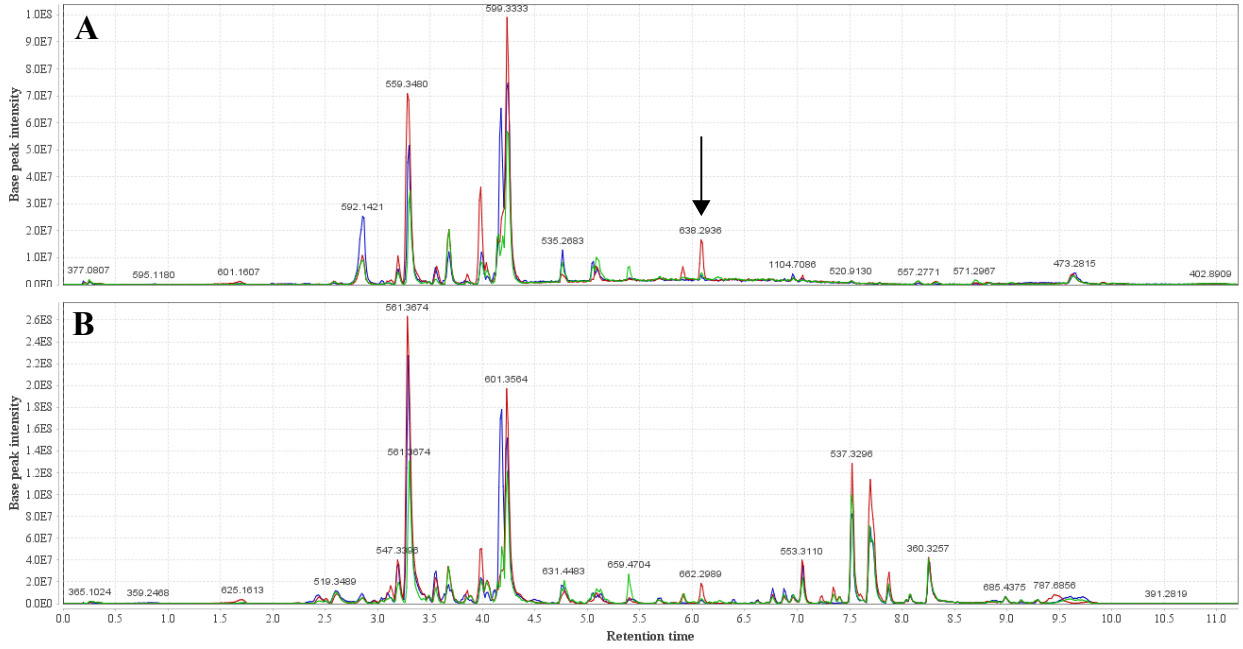


Figure 1.9: Metabolite profile comparison between *S. leeuwenhoekii* C34 (blue), M1600 (red) and M1601 (green). Negative ionization (**A**); positive ionization (**B**); the arrow indicates the peak corresponding to chaxamycin A (638.29 *m/z*) that was more intense in the mutant strain M1600.

Comparison of *S. leeuwenhoekii* C34, M1614, M1616, M1653, M1601 and M1615. Cultures in two production media, mDM and mLPM, were analysed using an ESI-IT Esquire 4000 (Bruker Daltonik GmbH, Germany). An ion of *m/z* 717.87 $[M + H]^+$ RT 40.6 min that has a chlorinated pattern was detected in the samples from mDM of *S. leeuwenhoekii* M1601 and M1615, and also in the samples from mLPM of *S. leeuwenhoekii* M1614 and M1615 (Figure 1.10). The ion 715.84 $[M - H]^-$, equivalent to the ion 717.87 $[M + H]^+$, was detected in the supernatant of *S. leeuwenhoekii* M1601 but with lower intensity. This ion was also detected in samples from *S. coelicolor* M1152 (data not shown).

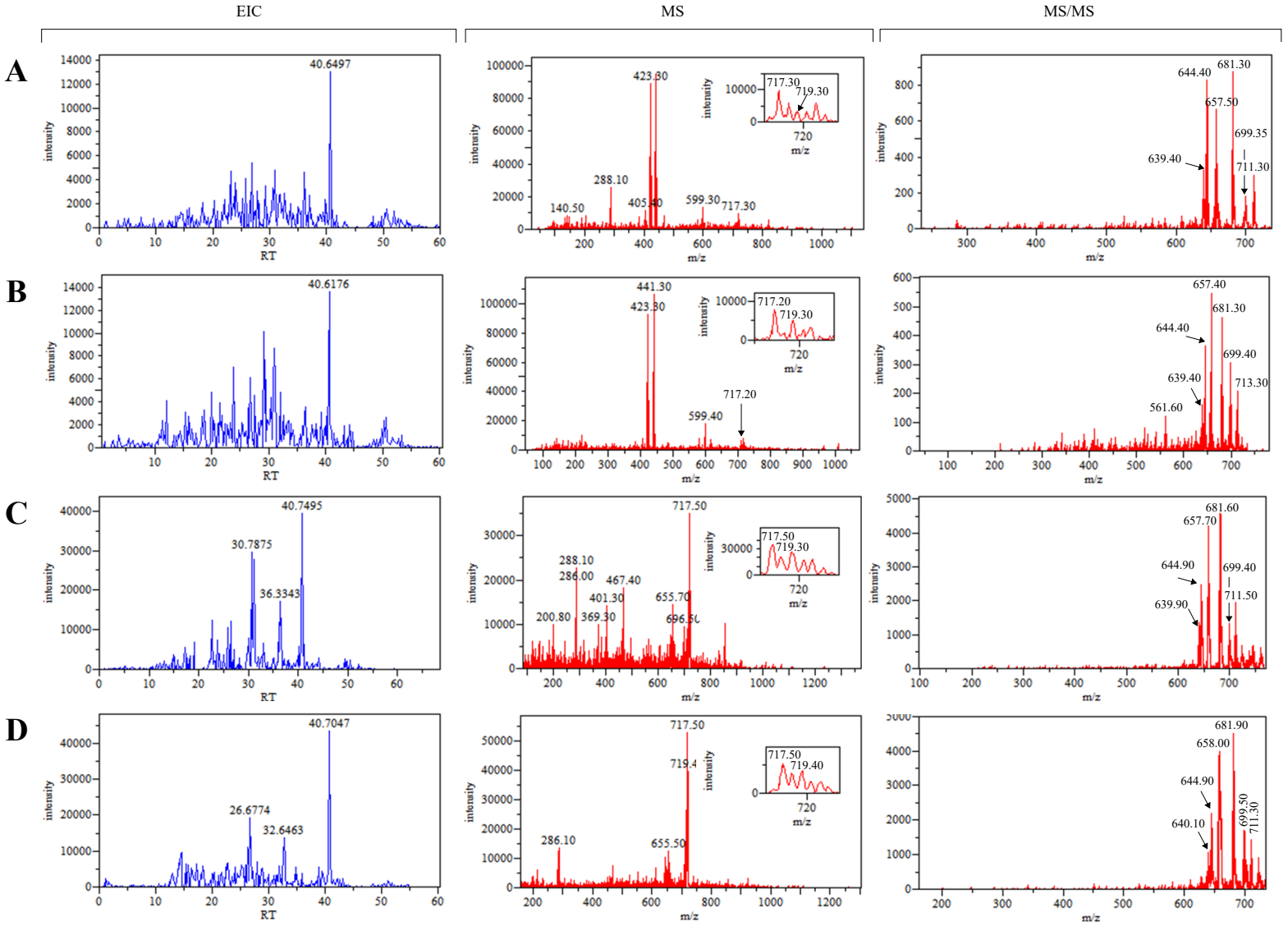


Figure 1.10: Comparison of fragmentation pattern of the ion *m/z* 717.87 $[M + H]^+$ RT 40.6 min found in the samples from supernatant of mLPM of *S. leeuwenhoekii* M1615 (**A**) and *S. leeuwenhoekii* M1614 (**B**), and also found in samples from supernatant of mDM of *S. leeuwenhoekii* M1615 (**C**) and *S. leeuwenhoekii* M1601 (**D**). Samples were analysed with an ESI-IT Esquire 4000 LCMS.

Comparison of *S. leeuwenhoekii* C34, M1614 and M1619. Seed cultures for *S. leeuwenhoekii* C34, M1614 and M1619 in mISP2 were carried out and continued with cultures in mLPM production medium. The samples from mycelium extract of two days old seed culture were used for HPLC MS/MS analysis since they show an interesting result in the bioassay (Section 1.4.7). In the sample from *S. leeuwenhoekii* M1619 a peak of *m/z* 609.77 at RT 45.33 min was found, that does not have a chlorinated pattern, however it was not observed in the other samples (Figure 1.11). The fragmentation pattern of the ion *m/z* 609.77 $[M - H]^-$ is shown in the Figure 1.12 **A**.

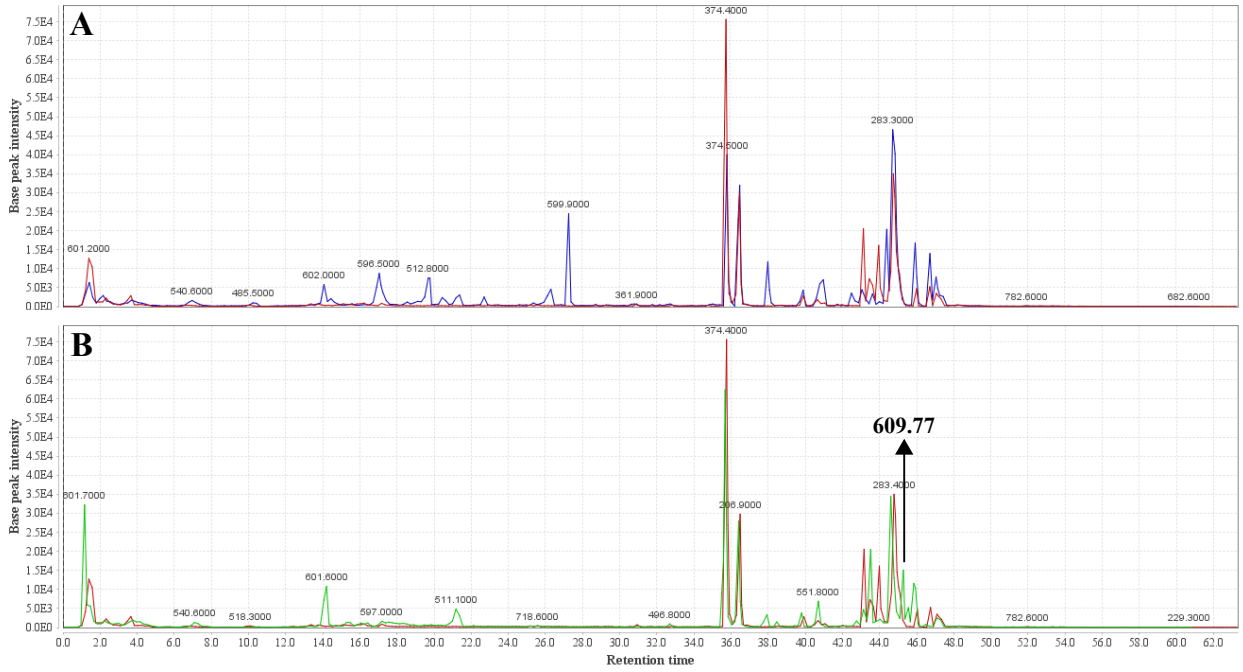


Figure 1.11: Metabolite profile comparison between *S. leeuwenhoekii* C34 (blue), M1614 (red) and M1619 (green) in negative ionization. Chromatogram comparison of *S. leeuwenhoekii* C34 and M1614 (**A**); chromatogram comparison of *S. leeuwenhoekii* M1614 and M1619 (**B**); the arrow indicates a peak corresponding to *m/z* 609.77, that was observed in the mycelium sample taken from *S. leeuwenhoekii* M1619.

Comparison of heterologous hosts containing the PAC clones. The heterologous expression of the hybrid PK-NP BGC was attempted to perform in three hosts: *S. coelicolor* M1152, *S. coelicolor* M1154 and *S. albus* J1074. Three media were tested: DM, R3 and TSB/YEME. Any distinctive peak on the samples containing the PAC clones was not found, neither any compound with a chlorinated pattern was found. The comparison between *S. coelicolor* M1152 and *S. coelicolor* M1607 (containing PAC 14B) is shown in the appendix information (Figure E.9). *S. albus* J1074 showed not to be a good heterologous host since several peaks were found in the WT strain.

Also, the heterologous host containing the overexpression of *sle09760* (*lys*R) did not show any increased specialised metabolite production, neither *S. leeuwenhoekii* M1605 (*sle09760-lys*R overexpression) (data not shown).

Comparison of *S. leeuwenhoekii* M1614 and M1619. A final comparison between these two strains was done using a Maxis II qTOF LCMS equipment and using different extraction methods. In the samples that were extracted with acetone it was observed more peaks than in the samples of

the same strain extracted with methanol (see supplementary Figure E.10)

The previously determined ion *m/z* 609.77 $[M - H]^-$ (Figure 1.11) was detected at RT of 8.96 min in the samples from *S. leeuwenhoekii* M1619 with *m/z* 609.51 $[M - H]^-$ and 611.53 $[M + H]^+$. It was confirmed by the fragmentation pattern of these metabolites that they corresponded to the same compound (Figure 1.12). As the equipment Maxis II qTOF LCMS has higher resolution and sensitivity it was assumed that the *m/z* 609.51 $[M - H]^-$ or *m/z* 611.53 $[M + H]^+$ was accurate. The intensity of this ion was greater in the samples of M1619, and it was detected easily in the positive ionization chromatogram.

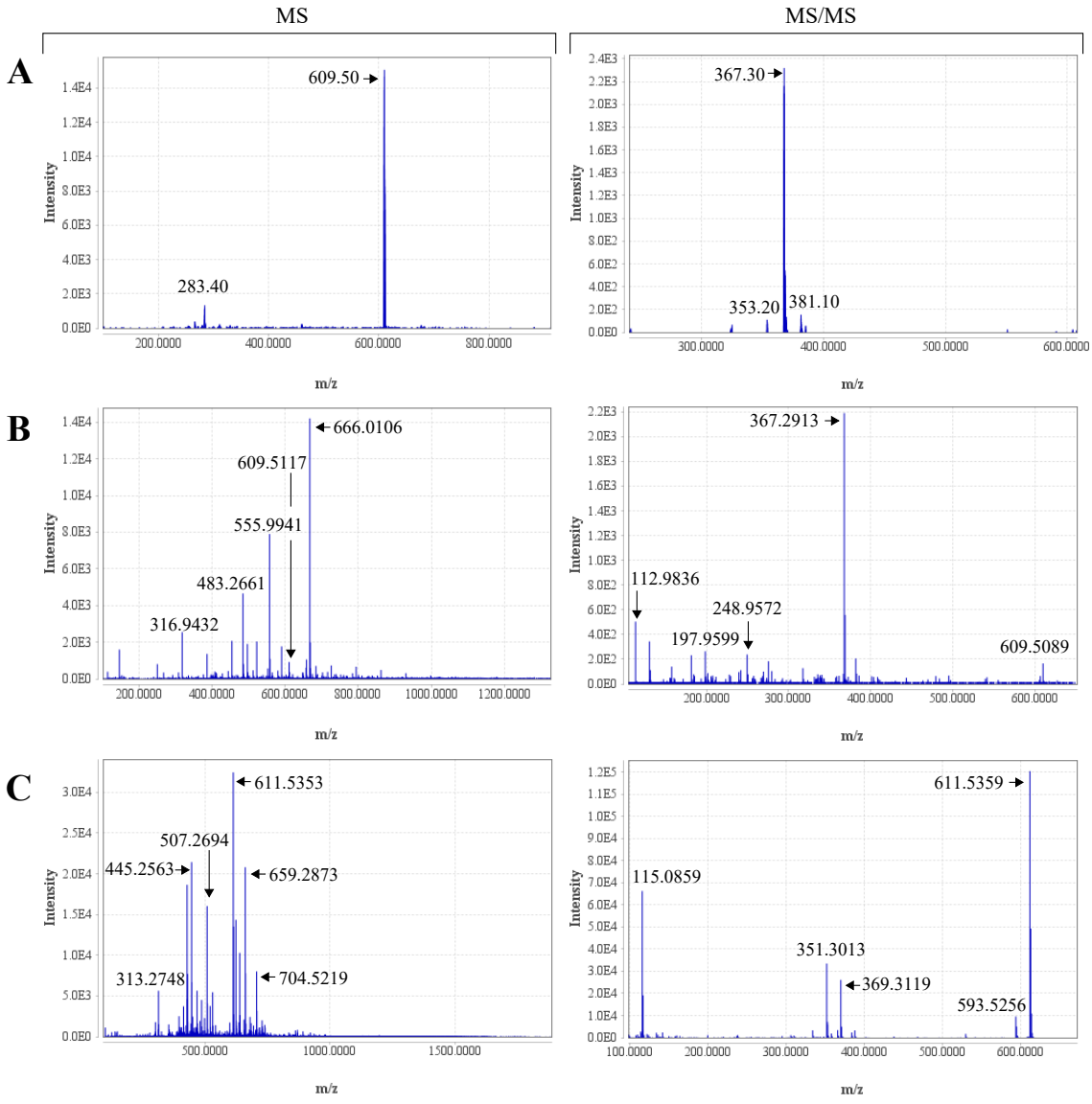


Figure 1.12: Fragmentation pattern of the ion of *m/z* 609.77 $[M - H]^-$ (**A**), *m/z* 609.51 $[M - H]^-$ (**B**) and *m/z* 611.53 $[M + H]^+$ (**C**), found in the sample of mycelium extract of 2 or 5 days old liquid culutre in mISP2 of *S. leeuwenhoekii* M1619, respectively. The sample of **A** was analysed with ESI-IT Esquire 4000 while the samples from **B** and **C** were analysed with a Maxis II qTOF LCMS

In the negative mode several chlorinated candidates were detected but the intensities were very low. On the other hand, when the *m/z* ratio of the ions was searched in the positive ionization

result, none of the previously *m/z* ratios were found. Search for these ions in the previous results was unsuccessful.

The ion *m/z* 717.87 $[M + H]^+$ or the equivalent in negative ionization was found as 717.19 $[M + H]^+$ or 715.16 $[M - H]^-$ at RT 5.21 min. Beside that ion, at the same RT other ions that have a chlorinated pattern: *m/z* 731.17, 747.16, 765.17, 781.17, 793.17 and 797.17 in the positive ionization mode were found (Figure 1.13); 729.14, 746.14, 763.15 791.14, in negative ionization (Figure 1.14). The peaks had more intensity in the positive ionization run, but the fragmentation pattern was just available in the negative ionization analysis for the peaks of *m/z* 715.16, 746.14 and 791.14 (see Figure E.11 in supplementary information). The MS/MS data obtained in the negative mode of the ion 715.16 was used to compare it with the previous fragmentation information (Figure 1.15).

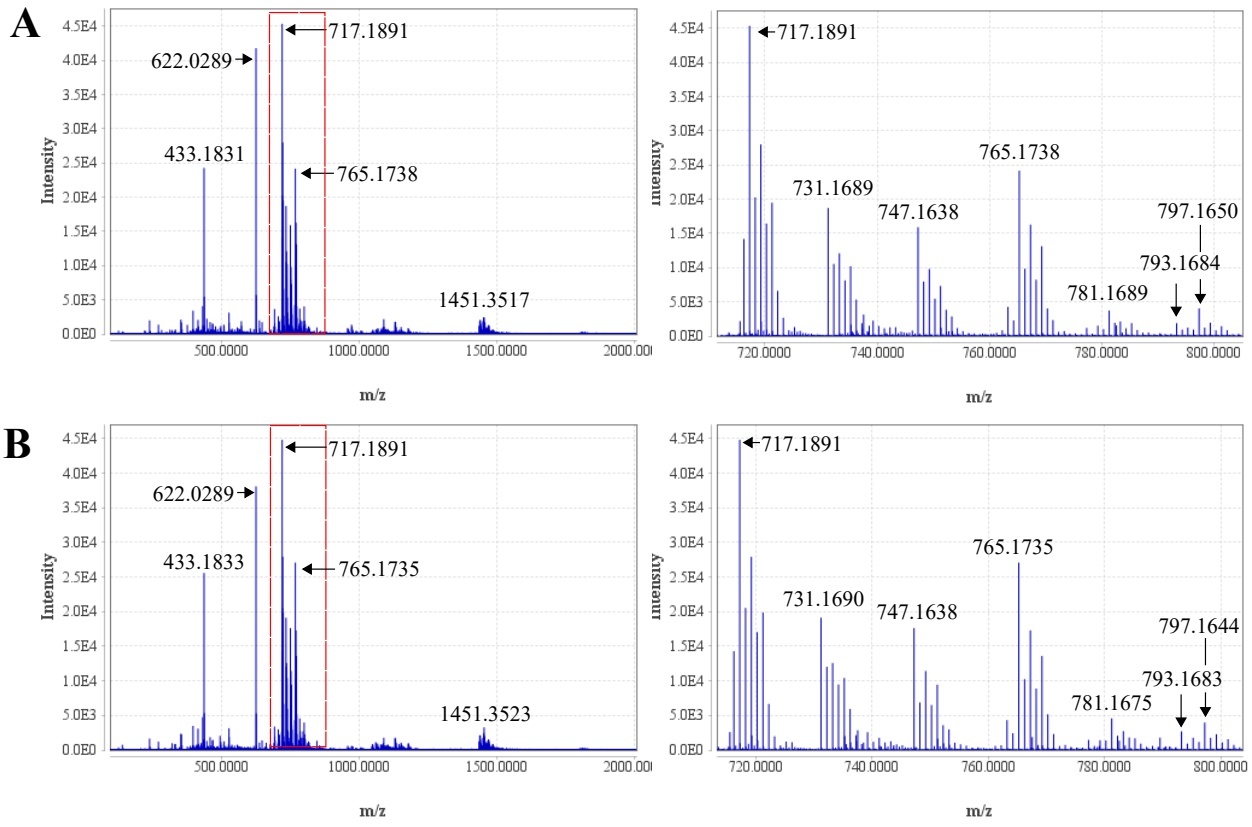


Figure 1.13: Mass spectrum at RT 5.2 min. of mix samples of supernatant and mycelium extract of *S. leeuwenhoekii* M1614 (**A**) and M1619 (**B**). The right graphs correspond to a zoom to the peaks of interest (red rectangle in the left graphs). The samples were run with positive ionization in a Maxis II qTOF LCMS. Mass spectrum was visualized with Mzmine (Pluskal et al., 2010).

1.4.7 Bioassays against *B. subtilis*

For all the strains generated in this work, bioassays were done in order to see if they increase or diminished the antibiotic activity against *B. subtilis*. In the bioassay of the heterologous host containing the PAC clones (18O or 14B) no antibiotic activity was found, and that is in accordance with the results of metabolic profile, where none distinctive peaks were found in the samples from

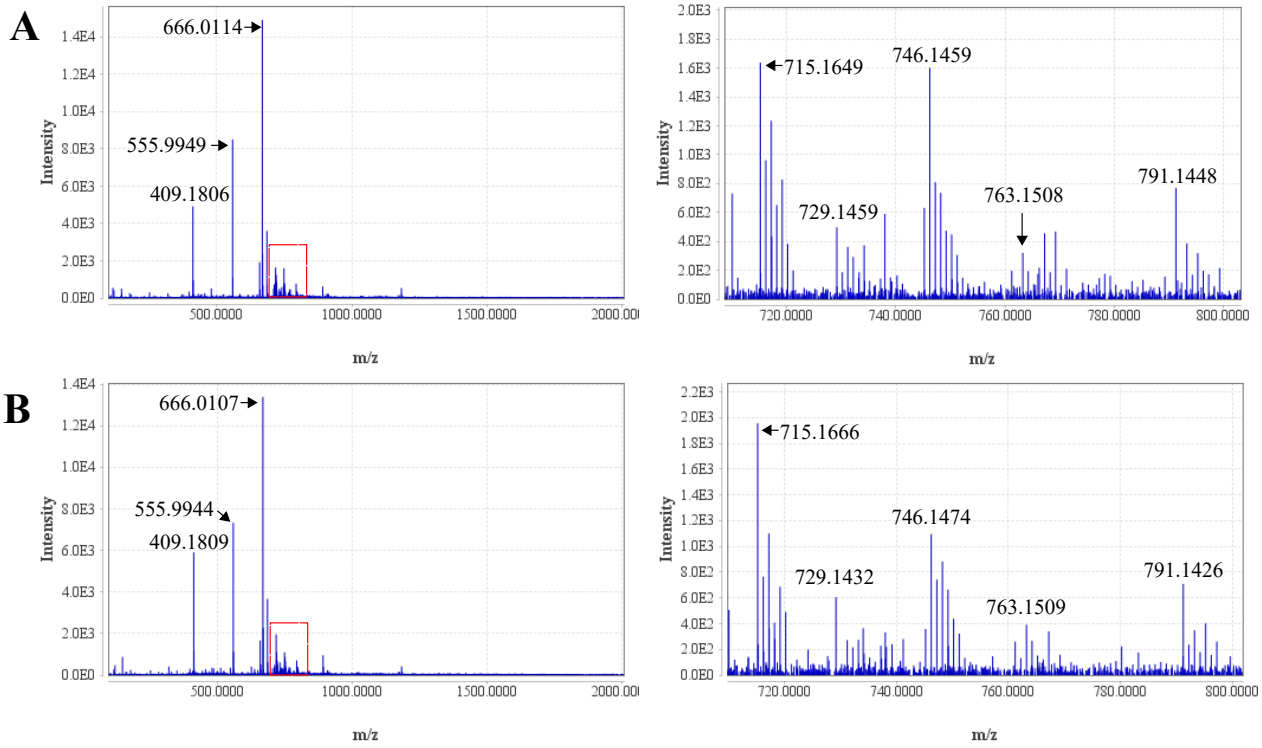


Figure 1.14: Mass spectrum at RT 5.2 min. of mix samples of supernatant and mycelium extract of *S. leeuwenhoekii* M1614 (**A**) and M1619 (**B**). The right graphs correspond to a zoom to the peaks of interest (red rectangle in the left graphs). The samples were run with negative ionization in a Maxis II qTOF LCMS. Mass spectrum was visualized with Mzmine (Pluskal et al., 2010).

the heterologous expression.

In Figure 1.16 and 1.17 are shown the result of bioassays. The plates were inoculated with *B. subtilis* and the samples of supernatant or mycelium extract were disposed in holes (for more details of the protocol used see Appendix D Section D.9). If the sample contained any compound with antibiotic activity against gram-positive bacteria it was possible to observe inhibition halos.

Mycelium extracts and concentrated supernatants from cultures in mDM of *S. leeuwenhoekii* M1614 did not show antibiotic activity, while the concentrated supernatant from mLPM showed little activity. The samples from *S. leeuwenhoekii* M1616 showed reduced antibiotic activity compared to *S. leeuwenhoekii* C34. On the other hand, samples from *S. leeuwenhoekii* M1601 and *S. leeuwenhoekii* M1615 showed equal or slightly higher antibiotic activity than the WT (Figure 1.16).

In the bioassay of the samples of two days old seed culture in ISP2 of *S. leeuwenhoekii* C34, M1614 and M1619, antibiotic activity was observed in the samples from C34 and M1619 (Δchaxamycin BGC, Δ*tetR*) but not in the sample from M1614 (Δchaxamycin BGC) (Figure 1.17). These samples were analysed by HPLC MS/MS and peak differences were found(see previous section). In the production media mLPM it was found that all of the samples have antibiotic activity (data not shown).

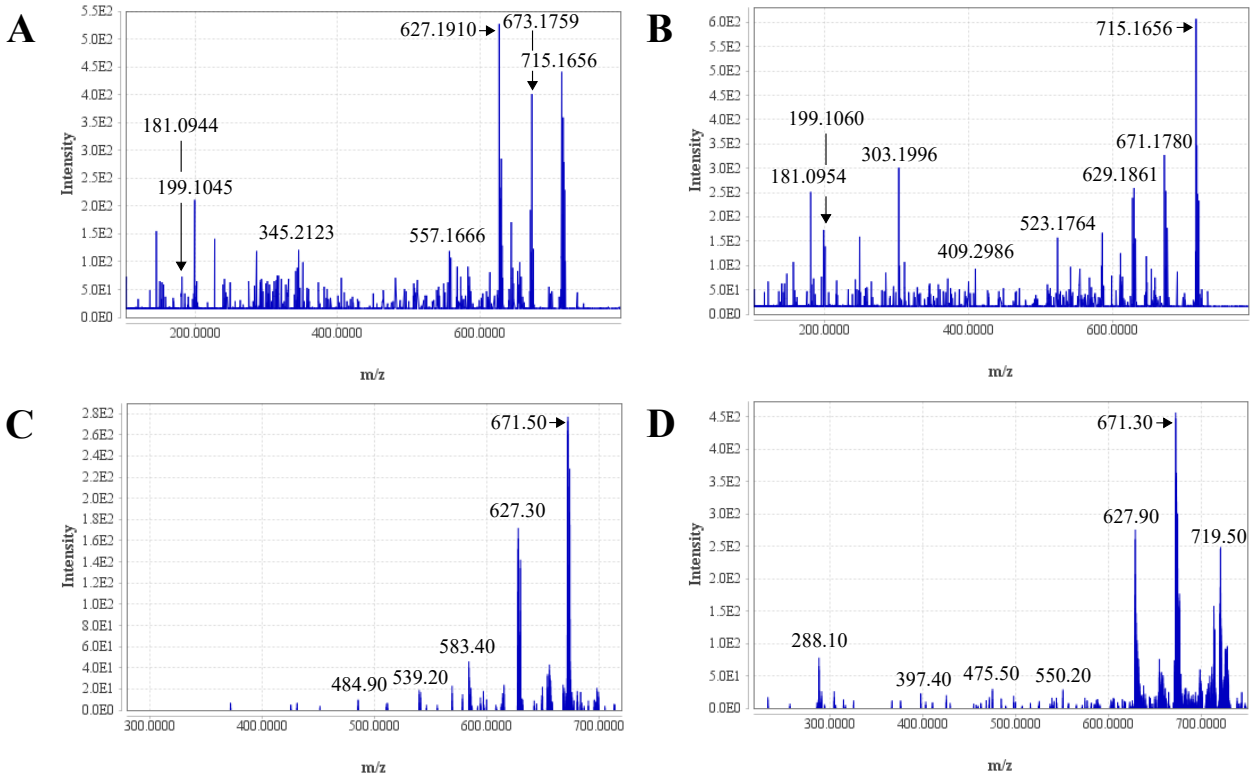


Figure 1.15: Comparison of fragmentation pattern of the ion *m/z* 715.16 [M - H]$^-$ RT 5.21 from sample of *S. leeuwenhoekii* M1614 (**A**) and *S. leeuwenhoekii* M1619 (**B**), with ion *m/z* 715.22 [M - H]$^-$ RT 40.92 from sample of *S. leeuwenhoekii* C34 (**C**) and with ion *m/z* 715.84 [M - H]$^-$ RT 40.82 from sample of *S. leeuwenhoekii* M1601 (**D**). **A** and **B** were obtained with a Maxis II qTOF equipment while **C** and **D** were obtained with a ESI-IT Esquire 4000.

1.4.8 Effect of increasing salt concentration in the production of specialised metabolites

The strains under salinity stress had to produce osmolytes to maintain the cell in positive water balance. These osmolytes or compatible solutes are of great biotechnology interest due to their used as stress-protective agents and stabilizers (Margesin and Schinner, 2001). One of the most abundant compatible solute in nature is ectoine (Margesin and Schinner, 2001), *S. leeuwenhoekii* C34 has the BGC of ectoine encoded in its genome.

Growth in increasing NaCl concentration in the culture medium from 0.2 % to 5 % w/v was evaluated. *S. leeuwenhoekii* C34 was able to grow at all tested concentrations however with more than 4 % of salt concentration the required lag phase was more than one week. Thus, samples of *S. leeuwenhoekii* C34 grew until 3 % NaCl were analysed by HPLC MS/MS using negative and positive ionization. In the comparison of metabolic profiles some peaks were found having higher intensities in some samples (Figure 1.18) but none of them look like a chlorinated compound in the positive ionization result. It should be noticed that in the negative ionization result there were some compounds with chlorinated profile but they were disregarded as being the compound of interest because they probably were chlorine adducts of other compound that eluted at the same RT. For example the metabolite of 693.44 *m/z* that had a chlorinated profile and probably was the chlorine adduct of the metabolite 657.46 *m/z* (Figure 1.18). Also, to confirm that this metabolite

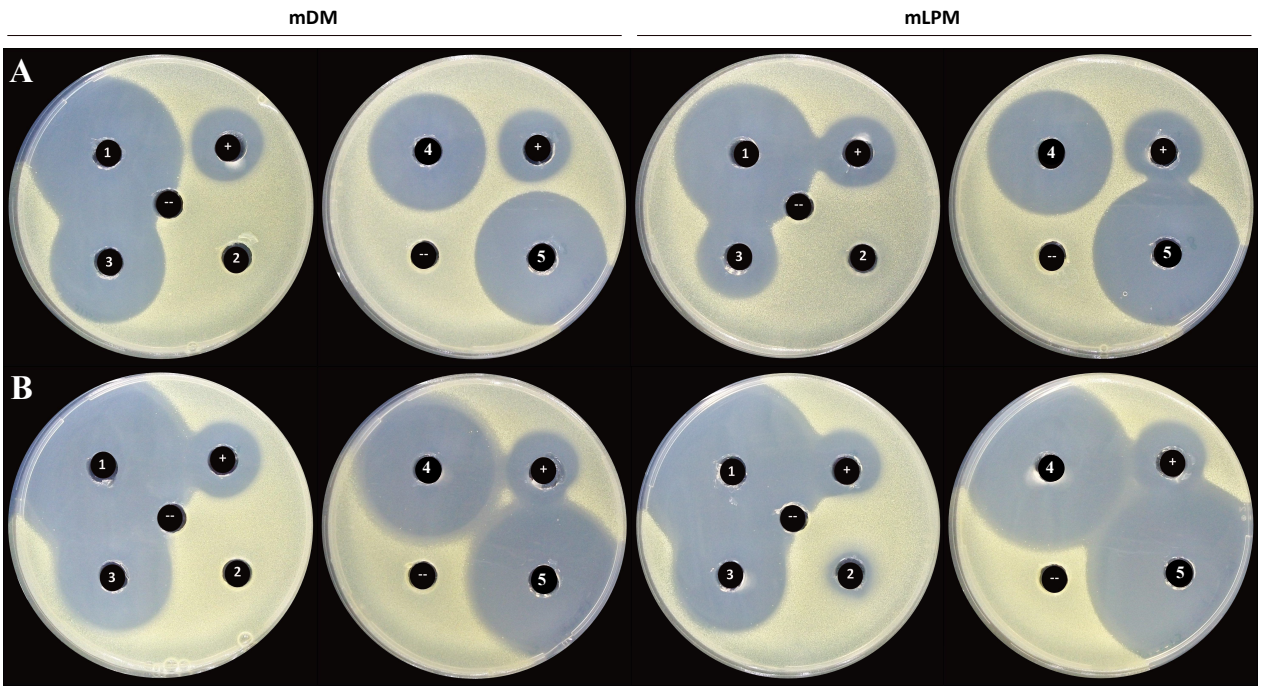


Figure 1.16: Comparison of bioactivity of samples of 5 days production culture of *S. leeuwenhoekii* C34 from mDM and mLPM against *B. subtilis*. Samples from methanol mycelium extract (**A**) and 10X concentrated supernatant (**B**). Samples: *S. leeuwenhoekii* C34 (1), *S. leeuwenhoekii* M1614 (Δchaxamycin BGC) (2), *S. leeuwenhoekii* M1616 (3), *S. leeuwenhoekii* M1601 (Δ*sle09500*) (4), *S. leeuwenhoekii* M1615 (Δ*sle09560*) (5). Control: 100 μg/ml apramacyn (+), methanol (-).

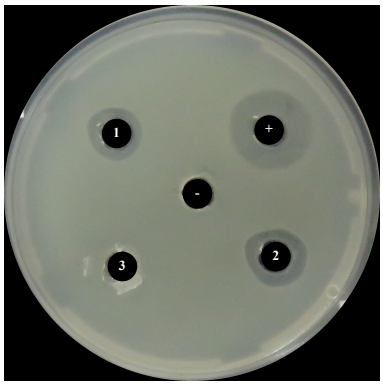


Figure 1.17: Comparison of bioactivity of methanol extraction of samples of 2 days seed culture in mISP2 against *B. subtilis*. Samples: *S. leeuwenhoekii* C34 (1), *S. leeuwenhoekii* M1619 (Δchaxamycin BGC; Δ*sle09560*) (2), M1614 (Δchaxamycin BGC) (3). Control: 100 μg/ml apramacyn (+), methanol (-).

was a chlorinated adduct, it was checked that it was also present in a sample of *S. leeuwenhoekii* M1600 (Δ*sle09470*). It was detected that the production of at least one type of chaxamycins and desferrioxamines was present in all the samples.

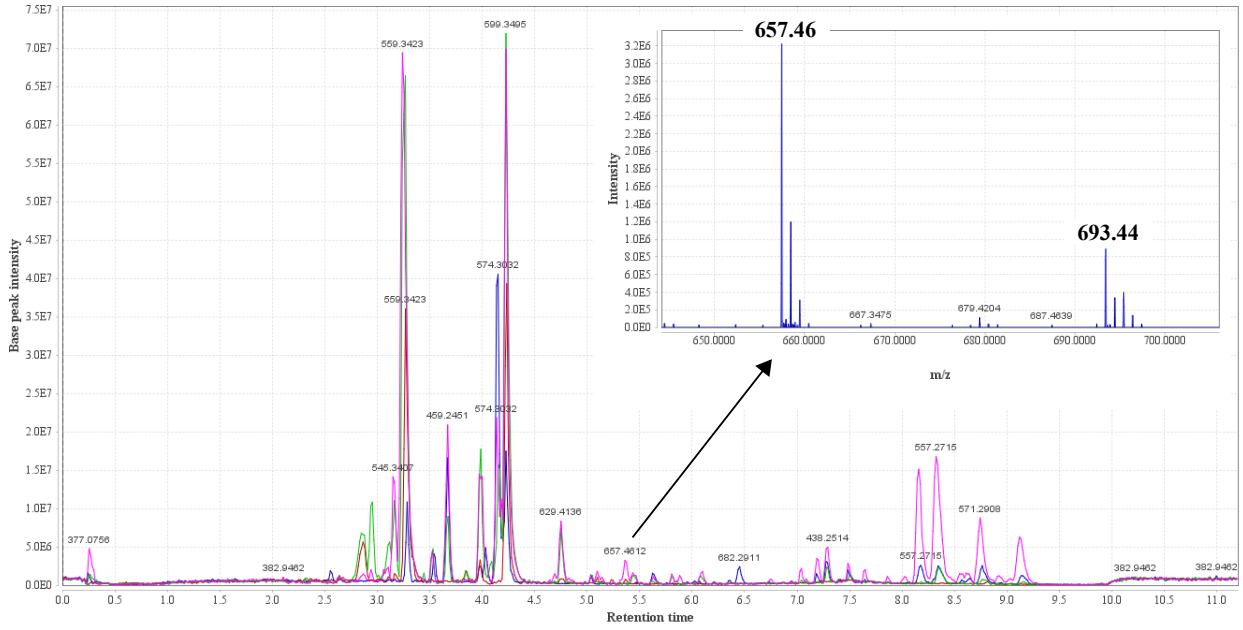


Figure 1.18: Metabolite profile of *S. leeuwenhoekii* C34 grew in presence of different salt concentration. The NaCl concentrations in (w/v) of each sample are: 0.2 % (blue), 1 % (pink), 2 % (green) and 3 % (red). The inset spectrum shows the *m/z* values at 5.4 min of retention time.

1.5 Discussion

1.5.1 The sequencing of *S. leeuwenhoekii* C34 genome allowed the identification of several biosynthetic gene clusters

When the genome of *S. leeuwenhoekii* C34 was sequenced de novo it was possible to identify 35 BGCs. Twenty-seven of the BGCs are probably producing new compounds, since those BGC have not been associated to known specialised metabolites. The number of BGC encoded in the genome of other *Streptomyces* strains is similar. For example, *S. avermitilis* has 30 BGCs (25 described by (Ōmura et al., 2001), plus 5 more described by (Ikeda et al., 2003)), *S. coelicolor* 18 BGCs (Bentley et al., 2002), *S. griseus* 34 BGCs (Ohnishi et al., 2008) and *S. davawensis* 32 BGCs (Jankowitsch et al., 2012).

1.5.2 The *trans*-AT PKS/NRPS BGC compared to the leinamycin BGC

In the particular case of the hybrid *trans*-AT PKS/NRPS BGC studied in this work, it was found that this gene cluster share similarities with the leinamycin BGC, a well studied hybrid *trans*-AT PKS/NRPS. There are three genes, *lnm*B, *lnm*C and *lnm*S, in the leinamycin BGC that does not have homologous genes in the hybrid *trans*-AT PKS/NRPS BGC. The gene *lnm*B encodes for a ferredoxin enzyme, the most similar protein in *S. leeuwenhoekii* C34 is Cxm21 (40 % of identity and 74 % of coverage) present in the chaxamycin BGC. The gene *lnm*C encodes for a hypothetical protein that does not have homologous in *S. leeuwenhoekii* C34. The gene *lnm*S encodes for an ABC transporter component, the function of this gene could be supplied by the gene *sle09300*.

1.5.3 Bioinformatic analysis of the trans-AT PKS/NRPS allows to predict the product of the BGC

The structure of type I PKS could be inferred from the gene sequences of the BGC. For example, chaxamycins are produced by a type I PKS and from the gene sequences it is possible to correctly predict the known structure of them (Castro et al., 2015; Castro, 2015). On the other hand, the prediction of the structure of *trans*-AT PKS are not so intuitive and they usually have special features, like domains of unknown function, repetitive domains and different assembly order (Piel, 2010; Alt and Wilkinson, 2015; Helfrich and Piel, 2016). Therefore the prediction of the structure based in the domain configuration and gene order could be wrongly interpreted.

Based in the available information of the leinamycin BGC and with bioinformatic analysis of specificity of the *trans*-AT and A domains, it was possible to predict the possible core structure of the hybrid PK-NP. Since several genes involved in β-branching were found in the hybrid *trans*-AT PKS/NRPS BGC it was included as information for prediction of the core structure.

A conserved motif that allow to identify ACP domains involved in producing β-branching has been described elsewhere (Haines et al., 2013). However the motif that was clearly identified in almost all the ACP associated to β-branching in that study was not found in the ACP that belongs to the hybrid *trans*-AT PKS/NRPS BGC. Also the ACP of leinamycin BGC did not have the conserved motif (Haines et al., 2013). So it is possible that at least the β-branching that has been described for leinamycin is being incorporated in the hybrid PK-NP.

1.5.4 Could the hybrid PK-NP be a halogenated compound?

It has been detected that this strain produces a chlorinated compound (M. Jaspars, personal communication) analysing the samples with ICP, a very sensitive technique for detecting metals. However it has been very hard to identify chlorinated compounds using HPLC MS/MS in this work. Nevertheless, the gen *sle09470* located in the hybrid *trans*-AT PKS/NRPS BGC encodes for a protein that could be a chlorinating enzyme, and it is the only putative chlorinating enzyme found in the *S. leeuwenhoekii* C34 genome associated to a BGC. Taking this information into consideration it is most probable that the hybrid PK-NP is the chlorinated compound detected previously.

Sle09470 is homologous to proteins described with chlorinating activity, specifically CmaB (NCBI accession number: AAC46036) have been described as the first of its type to have halogenase activity, and the activity was dependent of Fe^{2+}, α-ketoglutarate, O_2 and Cl^- (Vaillancourt et al., 2005). It is possible that all the requirements needed for the Sle09470 activity where not achieved and therefore the compound could not be chlorinated.

It is also possible that the compound is being produced in small quantities so it has not been possible to detect the compound in the analyses, also it could be that only the non-chlorinated version is being produced.

1.5.5 Why has it been so difficult to detect/identify the specialised metabolite produced by the hybrid *trans*-AT PKS/NRPS BGC?

There are several reasons why the product of the studied BGC has not been detected in the HPLC MS/MS analysis. But some of the main causes are the following:

The product of the BGC is being synthesised in very low concentrations, therefore it is not possible to detect it. This could be since the system to overcome the toxicity of the BGC product allows the production of low amount of the compound. The producer microorganisms had self-resistance systems in order to be able to produce toxic compounds (Hopwood, 2007), one mechanism is associated to the exportation of the metabolite (Pearson et al., 2004; Tahlan et al., 2007). In such cases, to achieve the production of a higher amount of the specialised metabolite, it is necessary to study the regulation of the genes associated with the transport of the metabolite. In the case of hybrid PK-NP there are transporter genes in the hybrid *trans*-AT PKS/NRPS BGC that could be in charge of the exportation of the compound. Specifically, the transporter gene *sle09570*, located downstream of a negative regulator *sle09560* (homologous to *tetR*), belongs to the major facilitator superfamily and could be functioning as a drug efflux pump.

The system TetR/TetA(efflux pump) had been studied in *E. coli* (Baumeister et al., 1992; Ramos et al., 2005). TetR represses *tetA* until tetracycline interacts with the TetR repressor generating a loss of affinity to the operator, this allows the transcription of *tetA* activating the resistance mechanism (Ramos et al., 2005; Hopwood, 2007). The homologous proteins, ActR/ActA, had been associated to resistance to actinorhodin. In this case, the protein ActR binds with higher affinity to a precursor of the actinorhodin biosynthesis rather than actinorhodin (Tahlan et al., 2008). This system allowed the microorganism to be prepared for the biosynthesis of the toxic compound (Hopwood, 2007; Tahlan et al., 2007, 2008).

In the particular case of the hybrid *trans*-AT PKS/NRPS BGC the ligand of *sle09560* (*tetR* homologue) is unknown. However, the ΔTetR strains developed in this work should have allowed a higher production of the hybrid PK-NP. Nevertheless, if the production of hybrid PK-NP continued to be low due to lack of transporter capacity, it would be necessary to induce the overexpression of the transporter (*sle09570*) by inserting a strong promoter such as *ermE* in front of the transporter gene (Medema et al., 2011b). Further analyses are needed in order to determine if the overexpression of this transporter in a ΔTetR strain, has a positive effect in the production of the hybrid PK-NP.

Another reason could be that the production of the BGC product by the microorganism is too expensive due to low availability of precursors. The precursors needed for the biosynthesis of many specialised metabolites are also required for the growth of the cell (Hiltner et al., 2015). Hence, to produce/over-produce a desired metabolite it is necessary to increase the availability of precursors. For example, the production of aromatic compounds in *E. coli* required erythrose 4-phosphate and phospho-enolpyruvate as precursors. Only when the enzymes involved in the production of these compounds are over-expressed is possible to increase the production of aromatic compounds to nearly the theoretical maximum yield (Patnaik and Liao, 1994). Therefore, it would be necessary to address genetic/metabolic engineering modifications in *S. leeuwenhoekii* C34 that would increase

the precursors supply, in order to improve the production of the specialised metabolites facilitating their detection.

The culture conditions studied were not optimal for the expression/detection of the BGC product. The technique called one strain many compounds (OSMAC), considers the use of different production media to accomplish the production of different specialised metabolites (Bode et al., 2002), since variation in the culture conditions could trigger the production of new compounds by mimicking an artificial environmental change (Gross, 2007). Even though a possible product of the BGC was detected in the DM used in this work, it could be that this medium was not the optimum for the production of the hybrid PK-NP. Accordingly, the study of changes in the culture conditions (temperature, pH, media composition, aeration rate, etc.) could be addressed.

Normal variation in the production of the specialised metabolites. It has been observed for this and other strains that a compound known to be produced by the microorganism is not generated even when the same conditions are used (Prof. M. Bibb, personal communication). This could be associated to small differences of physiological status of the strain, or others. This makes difficult the detection of the compound when doing metabolite profiling.

The chromatographic conditions used were not optimal. It is possible that the conditions used for the chromatographic analysis were not the best ones for the detection of the product of the hybrid *trans*-AT PKS/NRPS BGC. The specialised metabolites are structurally diverse and therefore different chromatographic conditions could be needed for their identification/detection (Frisvad, 1987).

Despite these difficulties, there are some potential candidates for the hybrid *trans*-AT PKS/NRPS BGC product but further studies are required in order to confirm or reject them (Section 1.5.8).

1.5.6 Utility of the antibiotic activity bioassay

The antibiotic activity bioassay used in this work, was used as an easy to do test to see if the compounds of interest have any antibiotic activity. However, since this strain produces several specialised metabolites with antibiotic activities (Rateb et al., 2011a,b) it is hard to realize which compound is responsible for the inhibition. In the majority of the cases it could be due to chaxamycins.

When the chaxamycins BGC was deleted from the genome of *S. leeuwenhoekii* C34 (strain M1614), in some of the bioassays it was possible to see the lack of antibiotic activity (either supernatant or mycelium extract). Nevertheless, sometimes the strain M1614 also had antibiotic activity. In a specific bioassay antibiotic activity with the strain *S. leeuwenhoekii* M1614 was not detected, but antibiotic activity was observed in the strain *S. leeuwenhoekii* M1619 (Figure 1.17).

On the other hand, the compound of interest could not have antibiotic activity at all, in which case this type of bioassay cannot help in the identification of the compound.

1.5.7 The heterologous expression of the hybrid *trans*-AT PKS/NRPS BGC did not work

The heterologous expression system of *S. coelicolor* has been used successfully with the study of other BGC (Gomez-Escribano and Bibb, 2014; Castro et al., 2015) and also for the lasso-peptide 3 BGC studied in this work. However it did not always work, and one of the reasons could be because the BGC needs other genes that were not included in the fragment selected and cloned, and there are not homologous genes in the host that could fulfil the required function. Even considering that the selected BGC boundaries for the screening of the PAC library included more than the genes predicted to be involved of the BGC, it could be that there are other genes in other sites of *S. leeuwenhoekii* C34 genome that are needed for the production of the compound. The other reasons are the same as those described in the section 1.5.5)

1.5.8 Possible candidate for the product of the hybrid *trans*-AT PKS/NRPS BGC

It is possible that the biosynthesis of the metabolite *m/z* 609.77 $[M - H]^-$ equivalent to 609.51 $[M - H]^-$, is associated with the deletion of the TetR regulator (*sle09560*), since it was first detected in a sample from *S. leeuwenhoekii* M1619 and in later analysis it was always found with more intensities in the samples from *S. leeuwenhoekii* M1619, rather than *S. leeuwenhoekii* M1614. It is also possible that the antibiotic activity observed in the sample from *S. leeuwenhoekii* M1619 (Figure 1.17), is due to this ion, since in that experiment it was not detected in *S. leeuwenhoekii* M1614 (strain that did not show antibiotic activity at that time point). Although this ion does not have a chlorinated pattern, so in the case that is a product of the hybrid *trans*-AT PKS/NRPS BGC, it would not be the chlorinated version. Sadly, this compound was not detected in previous assays when samples of *S. leeuwenhoekii* C34, *S. leeuwenhoekii* M1601 (Δ*trans*-AT) and *S. leeuwenhoekii* M1615 were analysed. So, to confirm that this compound is a product of the hybrid *trans*-AT PKS/NRPS it is necessary to perform the deletion of *sle09500* (*trans*-AT gene) in the strains of *S. leeuwenhoekii* M1614 and *S. leeuwenhoekii* M1619, and perform a metabolic profile comparison.

Using the information obtained from the structure prediction of the hybrid *trans*-AT PKS/NRPS BGC product it is possible to use ICP-MS with more efficiency, since it could be used to search for molecules for Cl and S atoms (Prof. M. Jaspars, personal communication). This technique had been successfully used to determine halogen elements in organic compounds (Bu et al., 2003). The higher sensitivity of this techniques would allow to identify the halogenated product of the hybrid *trans*-AT PKS/NRPS BGC even if it is been produced in low quantities. On the other hand, over-expression of halogenating genes had proved to be useful for detecting chlorinated compounds in *S. venezuelae* (Thanapipatsiri et al., 2016), hence the overexpression of *sle09470* should be addressed in *S. leeuwenhoekii* M1614 and *S. leeuwenhoekii* M1619.

The ion *m/z* 717.87 $[M + H]^+$ had a chlorinated pattern, but it is unlikely to be the hybrid PK-NP because it was detected in the sample from *S. leeuwenhoekii* M1601 (Δ*trans*-AT) and also in samples from *S. coelicolor* M1152.

1.5.9 Metabolic profiling as a tool to find new specialised metabolites

When doing metabolic profiling it is important to try to use always the same conditions, equipment, column, time, type of sample, so the differences it the profile could be related to the different strains. However during the course of this work different equipment were used what makes difficult to make a comparison using all the collected data. For example, when an electrospray-ion tramp ESI-IT Esquire 4000 equipment was used some specific peaks were found, but the lack of resolution made it hard to identify the same compound in other samples, particularly in samples from equipment with higher resolution. This happened because when the samples were analysed with this equipment, not only the retention time was different, but also the relative *m/z* of the ions, for example chaxamycin A was found as *m/z* 638.8 (confirmed by fragmentation pattern analysis) and not always the fragmentation pattern was available.

When searching for a chlorinated compound it would be easy to analyse the positive ionization result because when the negative ionization is used, there is a possibility to obtain the chlorinated adduct of the compound that could lead to miss-interpretation of the result, especially when the sample used could have chlorine.

Regardless the difficulties found during this work, metabolic profile comparison had been successfully applied for the identification of novel compounds (Chooi et al., 2013; Djinni et al., 2014; Thanapipatsiri et al., 2016). Particularly, using this tool it was possible to identify the ion *m/z* 611.53 $[M + H]^+$ as a possible product of the studied BGC.

1.5.10 Future work

A GSM of *S. leeuwenhoekii* C34 will be used to predict modifications (gene deletions and overexpressions) that could enhance the production of the hybrid PK-NP, along side with the generation of mutants like Δ*trans*-AT in *S. leeuwenhoekii* M1614 and *S. leeuwenhoekii* M1619.

A recently developed software for automatic detection of chlorinated compounds from mass spectra data (Roullier et al., 2016) was applied to some of the samples obtained in this work. This software functions in R environment and is very easy to execute. Several halogenated candidates were detected according to the software but deeper analysis showed that a lot of them were false-positives. Some true-positives were identified but they have low intensities, further analysis will be done of all the possible candidates.

The deletion of the biosynthetic gene *sle09500* and overexpression of *sle09470* (encoding for a chlorinating enzyme) would be accomplish in the mutant strains *S. leeuwenhoekii* M1614 and *S. leeuwenhoekii* M1619. The technique ICP-MS would be used to study the production of halogenated compounds by the different strains generated during this work and with the strains that would be generated in the future.

On the other hand, other BGC encoded in *S. leeuwenhoekii* C34 genome, like the one encoding for a NRPS, or PKS, would be studied by cloning them in heterologous hosts, and performing

metabolic profile comparison.

1.6 Conclusions

1. *S. leeuwenhoekii* C34 has 35 putative gene clusters encoding for specialised metabolites and 28 are probably encoding for new metabolites.

2. Three lasso-peptides BGC were found in the genome of *S. leeuwenhoekii* C34. Lasso-peptide 3 was detected in the supernatant of liquid cultures, despite the absence of a transporter within the BGC. Also, it was identified as the same described in a closely related strain *S. leeuwenhoekii* C58.

3. A 64 kb BGC encoding for a hybrid *trans*-AT PKS/NRPS that spans from 1083651 to 1147687 nt was identified.

4. An ion *m/z* 611.53 $[M + H]^+$ was detected as possible product of the hybrid *trans*-AT PKS/NRPS BGC in *S. leeuwenhoekii* M1614 and with more intensity in *S. leeuwenhoekii* M1619.

Chapter 2

Analysis of metabolic networks of *Streptomyces leeuwenhoekii* C34 by means of a genome scale model: prediction of modifications that enhance the production of specialised metabolites.

2.1 Abstract

Specialised metabolites are produced by microorganisms using precursors synthesised during primary metabolism. Genome scale models (GSMs) are representations of the metabolism of an organism and allow to study the intracellular metabolic fluxes. In this work a GSM for *S. leeuwenhoekii* C34 was developed in order to study the biosynthesis pathways of specialised metabolites. The construction of the model, *i*VR1007, was done using the high quality genome sequence of *S. leeuwenhoekii* C34 and the available information of Gene-Protein-Reaction (GPR) of *Streptomyces* strains. To do so, a python-based interface was developed, that enables: a search for the *Streptomyces* genes associated to a reaction in the Kyoto Encyclopaedia of Genes and Genomes (KEGG) database, performance of local BLAST against a *S. leeuwenhoekii* protein database, comparison of the domain of the proteins, downloading of the metabolite information, construction of the GSM and performance of flux balance analysis (FBA) simulations using COBRApy. The biosynthesis pathways of specialised metabolites such as chaxamycins, chaxalactins, desferrioxamines, ectoine and the putative product of a hybrid *trans*-AT PKS/NRPS (hybrid PK-NP) were included in the model. The model *i*VR1007 consist of 1722 reactions, 1463 metabolites and 1007 genes, and it was validated using experimental information of growth in different carbon, nitrogen and phosphorous sources, showing a 83.7 % accuracy. The model was used to predict metabolic engineering targets for enhancing the biosynthesis of chaxamycins, chaxalactins and hybrid PK-NP. Gene knockouts that enhance the production of the specialised metabolites by increasing the pool of precursors were identified, such as *sle03600*, *sle17360* and *sle39090*. On the other hand, using the

algorithm of flux scanning based on enforced objective Flux (FSEOF) implemented in python, 35, 25, 27 over-expression targets for increasing the production of chaxamycin A, chaxalactin A, and the hybrid *trans*-AT PKS/NRPS product, respectively, that were not directly associated with their biosynthesis routes were found. Ten over-expression targets that were common to the three specialised metabolites studied, like the over expression of the acetyl carboxylase complex (*sle47660* and any of the following genes: *sle59710* or *sle44630* or *sle39830* or *sle27560*) were identified. The predicted knockouts and overexpression targets will be used to perform metabolic engineering of *S. leeuwenhoekii* C34 and obtain overproducer strains.

2.2 Introduction

The metabolism of microorganisms has been traditionally divided into primary and secondary metabolism. The primary metabolism is common to all microorganisms and is integrated by all the essential reactions that allow growth. On the other hand, the secondary metabolism is thought to include species-specific pathways. During the secondary metabolism, specialised metabolites are synthesised using precursors synthesised in the primary metabolism (Hiltner et al., 2015), these metabolites accomplish diverse roles within the microbial environment such as defence, communication, metals transporters, sexual hormones and as differentiation effectors (Demain and Fang, 2000). Specialised metabolites have shown to have diverse chemical structures and many different applications, especially in medicine (McMurry and Begley, 2005; Gross, 2007).

The discovery of the first specialised metabolite with antibiotic activity, penicillin, was done by Alexander Fleming (Fleming, 1929) and it was carried to mass production by Florey and Chain in the 40's and since then a golden era (1940-1962) of antibiotics began (Singh and Barrett, 2006). During this time, several new compounds with interesting activities were discovered (Davies and Davies, 2010). However after a few decades, the rediscovery of the same compounds and antibiotic resistance become a recurrent issue.

The majority of the specialised metabolites with antibiotic activity came from *Streptomyces* strains (Kieser et al., 2000). The sequencing of *S. coelicolor* genome has allowed to study the biosynthesis of specialised metabolites and also to identify silent or cryptic biosynthetic gene clusters (BGCs) encoding for new compounds (Lautru et al., 2005; Challis, 2008b).

Nowadays the need for new specialised metabolites, especially antibiotics is urgent. Understanding the connection between primary and secondary metabolism, to study the pathways of precursors biosynthesis and to exploit the genomic potential of the microorganisms is essential in order to produce them and improve their yield.

A way of doing this is through a GSM. GSMs are mathematical models that are developed to study the behaviour of the cell taking into account all the available information of the genome, omics data and literature. Since the development of the first GSM in 1999, there have been several genome scale models for different strains (Kim et al., 2012). These models have allowed determining the behaviour of the cell under various conditions simulating genes deletion or over-expression and their effect in the organism studied and identifying drug targets amongst other things (Cam-

podonico et al., 2014; Contador et al., 2015). Specifically for *Streptomyces* strains, GSM have been developed for *S. coelicolor* (Kim et al., 2004; Borodina et al., 2005; Alam et al., 2010; Kim et al., 2014), *S. clavuligerus* (Medema et al., 2010), *S. lividans* (D'Huys et al., 2012) and *S. tsukubaensis* (Huang et al., 2013). These models have been used to predict gene targets for metabolic engineering in order to improve specialised metabolites production, among other things. For example, the GSM of *S. tsukubaensis* was used to predict gene knockouts and overexpression that enhance the production of the immunosuppressant FK506.

In this thesis, the first GSM for *S. leeuwenhoekii* C34 was constructed using the genome sequence information obtained previously (Gomez-Escribano et al., 2015). The GSM was used to study precursors biosynthesis of specialised metabolites, such as chaxamycins, towards improving their yield. Also, identification of gene targets for overexpression or deletion that would enhance specialised metabolites biosynthesis is addressed.

2.3 Methodology

2.3.1 Bacterial strains

A list of the strains used in this work is available in Appendix A Table A.1. *S. leeuwenhoekii* C34 was isolated from the Atacama Desert by a collaboration with Prof. Alan Bull (University of Kent) and Prof. Michael Goodfellow (University of Newcastle).

2.3.2 Sole carbon sources assay

For one sole carbon source experiment a minimum agar media was used (Pridham and Gottlieb, 1948), and the corresponding carbon source was added to a 1% w/v final concentration. Plates with the carbon source were inoculated with 20 μl of *S. leeuwenhoekii* C34 spore stock, that was dispersed with the aid of an sterile cotton bud through all the plate. The plates were incubated at 30 °C and evaluated for growth at 7, 14 and 21 days.

2.3.3 Reconstruction of the genome scale model

The methodology for the reconstruction of the GSM of *S. leeuwenhoekii* C34 is summarized in Figure 2.1. Concisely, the information of GPR of other *Streptomyces* especially from *S. coelicolor*, enzyme commission (EC) numbers and pathways was downloaded from KEGG database (Kanehisa and Goto, 2000; Kanehisa et al., 2014) using the bioservices module (Cokelaer et al., 2013) with a script written in python programming language version 2.7 (python programme available at https://www.python.org/download/releases/2.7/). All the genes of *S. coelicolor* associated to a metabolic pathway were retrieved and using the orthology numbers information the reactions were designated. Later, information of genes encoding for proteins absent in *S. coelicolor* was downloaded from other *Streptomyces*. Using their orthology number the information of the reactions was retrieved. For transport reactions, information of the associated genes was obtained from ei-

ther KEGG or transportDB (Ren et al., 2007).

Local blast of those genes of *S. coelicolor* or another *Streptomyces* was done against a local *S. leeuwenhoekii* C34 database. A threshold of at least 45 % identity and e-value lower than 0.001 was defined as a cut-off to select for the best hits. On the other hand, CD-blast (Marchler-Bauer and Bryant, 2004; Marchler-Bauer et al., 2009, 2011, 2014) was performed separately for *S. leeuwenhoekii* C34 and for the other *Streptomyces*. The CD-blast information was used to compare the domains of homologous proteins to confirm their similarity in functionality.

The model was refined with bibliographic information. Specialised metabolites biosynthesis pathways were incorporated (for more details go to section 2.3.5). Artificial reactions for the biosynthesis of biomass were included in order to simulate cell growth (for more details go to section 2.3.4).

Manual curation of the designated GPR association was done for all the metabolic pathways of *S. leeuwenhoekii* C34. It was checked that the production of biomass was possible by evaluating the precursors supply to each building block of the biomass reaction. Gaps were incorporated if needed, especially when all the genes of the pathway down the gap were present in *S. leeuwenhoekii* C34. Charge and mass balances were checked.

The predictability of the model was assessed by comparison of in-silico results with experimental data (for more details go to section 2.3.8).

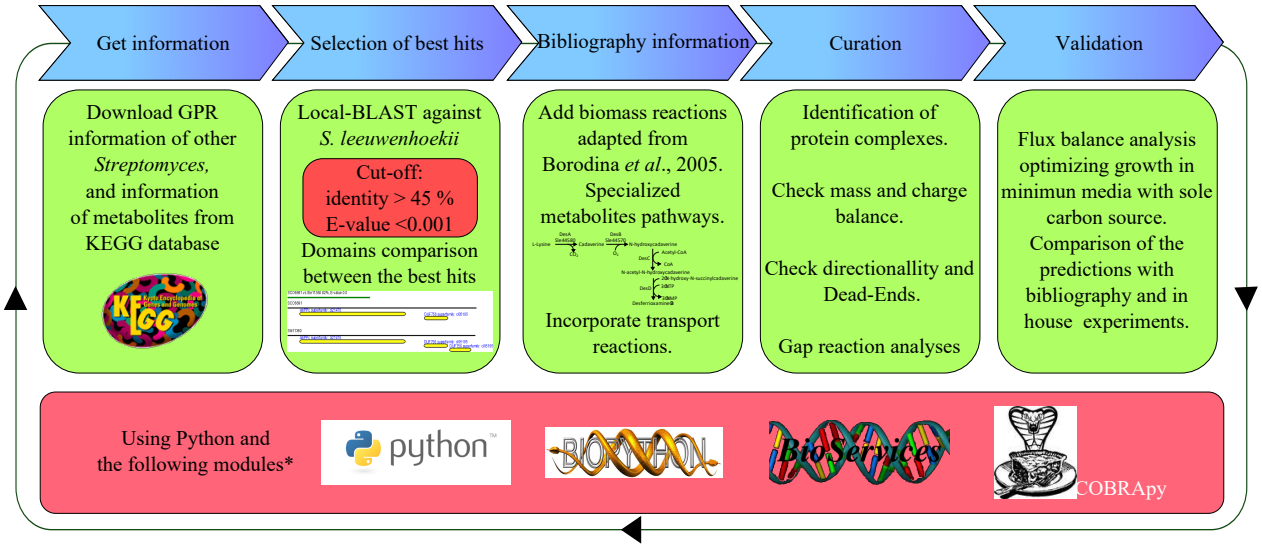


Figure 2.1: Methodology used for the reconstruction of the genome scale model of *Streptomyces leeuwenhoekii* C34. *Several other modules were also used during this methodology, such as Re, Matplotlib and Tkinter.

2.3.4 Biomass composition

The biomass elementary composition was adapted from the one described for *S. coelicolor* (Borodina et al., 2005), using information of *S. leeuwenhoekii* C34 when it was available and comple-

menting with information of other *Streptomyces* (for a detailed description please go to Appendix F.1). Briefly, the DNA composition was obtained from the deoxyribonucleoside composition of the chromosome of *S. leeuwenhoekii* C34. RNA composition was derived from the deoxyribonucleotide composition of rRNA, tRNA and mRNA, the later one inferred from the DNA sequence of the chromosome. Energy requirement for polymerisation of triphosphates was obtained from Ingraham et al. (1983). Composition of proteins and small molecules was assumed as for *S. coelicolor*. Triacylglycerols (TAGs) and lipids proportion of DW was taken from *S. coelicolor*, but TAGs and lipids composition was taken from *S. leeuwenhoekii* C34 (Busarakam, 2014).

2.3.5 Incorporation of specialised metabolites pathways

Biosynthesis pathways to produce chaxamycins (Castro et al., 2015), chaxalactins (Castro, 2015), desferrioxamines (Barona-Gómez et al., 2004) and the putative product of the hybrid *trans*-AT PKS/NRPS BGC (hybrid PK-NP), were manually included into the model. All the reactions needed for the biosynthesis of the precursors of the specialised metabolites were incorporated into the model. It was checked that the production of each specialised metabolites included was possible by setting the biomass production rate to a 10 % of the growth in default conditions and optimizing the specialised metabolites biosynthesis.

2.3.6 Curation of the model

The model obtained was manually curated and it was evaluated that the production of biomass was possible evaluating each precursor at the time. The directionality of the reactions was checked according to ΔG energies (Flamholz et al., 2011) and with bibliographic information of reversibility available of the closest phylo-genetically related microorganisms.

Furthermore, the predicted domains through conserved domain database (CDD) batch analysis for all the proteins related to reactions within the model reconstruction were compared to *Streptomyces* homologous proteins assigned to each reaction in KEGG. To do so, a python script that compares the domains of the *S. leeuwenhoekii* C34 proteins and other *Streptomyces* was developed. Within this script it was possible to obtain an image of the proteins that did not match in order to facilitate further analyses.

Also, it was checked that the domains of the proteins assigned to each reaction had a coherent functionality. In addition, genes encoding to protein complexes or isoenzymes were identified, and properly annotated.

2.3.7 Simulations of the metabolism

Simulations were done by FBA or MOMA using the COBRApy toolbox (Ebrahim et al., 2013) in python programming language version 2.7 (https://www.python.org/download/releases/2.7/). The GSM was represented as a stoichiometric matrix S of size $m \times n$, where m represents the number

of metabolites and *n* the number of reactions. Under steady state assumption the system of linear equations of mass balance is defined by:

$$Sv = 0 \tag{2.1}$$

where, *S* is a stoichiometrix matrix and *v* is a vector of length *n* that represents the flux through all the reactions.

The solution space was constrained by the boundaries of each reaction (Equation 2.2), and for the definition of an objective function *Z* (Equation 2.3). For reversible reactions the lower and upper bound were set as -1000 and 1000 respectively, leaving the reaction practically unconstrained, while for irreversible reactions the lower bound was set to zero. For simulation of specialised metabolites production, a percentage of the maximum biomass production rate calculated was added as a constraint, and a specific specialised metabolite was set as a new objective function *Z*.

$$v_{LB} \leq v \leq v_{UB} \tag{2.2}$$

$$Z = c^T v \tag{2.3}$$

The default carbon, nitrogen, phosphorous and sulphur sources were glucose, ammonium, phosphate and sulphate, respectively. Default conditions for simulating growth were set at -10 mmol gDW^{-1} h^{-1} uptake rates (lower bound), where a minus sign represents that the metabolite is being consumed. To simulate aerobic growth the oxygen uptake rate was set at -10 mmol gDW^{-1} h^{-1}. Small inorganic ions needed for biomass biosynthesis were allowed to freely enter or exit the system by setting the lower and upper bound of each exchange reaction to -1000 and 1000, respectively. The lower bound of the rest of exchange reactions was constrained to zero when simulating growth in MM. When simulating growth in MM with other carbon, nitrogen or phosphorous sources, the lower bound of the default source was set to zero, and the uptake rate of the evaluated source was set to -10 mmol gDW^{-1} h^{-1}.

Gurobi optimizer version 6.5.1 (http://www.gurobi.com/), under a free academic license, was used as the linear programming solver.

2.3.8 Validation of the model

Experimental results of growth or no-growth under sole carbon, nitrogen or phosphorous sources obtained in this work and obtained using Biolog (http://www.biolog.com/) by Dr. Castro in Newcastle, UK, was used to study the predictability of the model. The uptake rates of the exchange

reaction of each carbon, nitrogen or phosphorous source was set to -10 mmol gDW^{-1} h^{-1} one at a time. A viability threshold of 10 % of growth of default conditions was considered.

2.3.9 Gene knockout analysis and experimental studies

FBA and MOMA were used to simulate gene knockout and find which gene deletion could enhance synthesis of precursors towards specialised metabolites biosynthesis.

In order to simulate the gene deletions, simulation of growth in complex media was used as default condition. The optimization was carried out with the biomass production rate as objective function, and the initial values of production of chaxamycin A, chaxalactin A and hybrid PK-NP were obtained. Then the gene(s) associated to each reaction was knocked-out and it was evaluated if the deletion of the gene(s) increased the initial production rate of each evaluated specialised metabolites. The gene(s) knockout that generated the higher increase in the production, were used as genetic background for the search of double or triple gene knockouts that would further improve the yield of chaxamycin A, chaxalactin A or hybrid PK-NP. To ensure the viability of the cell, each knockout was selected considering a viability threshold of at least 10 % of the maximum growth in default conditions.

2.3.10 Identification of gene's overexpression targets

The algorithm FSEOF was used to select gene targets for overexpression (Choi et al., 2010). Through FSEOF it is possible to identify the fluxes that are increased at the same time the flux through a objective reaction is incremented, and the production of biomass is used as an objective function. The algorithm to carry on FSEOF (Choi et al., 2010) was implemented in python and used in conjunction with the COBRApy toolbox (to see details of the script used, go to Appendix F Section F.3.11). Since the FBA result is not unique, flux variability analysis (FVA) was applied to determine if the predicted targets for overexpression were true positives.

2.3.11 Implementation and software usage

The genome scale model was constructed using biopython and implemented using COBRApy (Ebrahim et al., 2013) installed in a computer with 64-bit Windows 10 system, Intel® Core^{TM} i5-2430M CPU @ 2.40 GHz with 8 GB RAM. General draws and figures were done using Inkscape (https://inkscape.org) or python. Draws of fluxes were done with Escher (King et al., 2015) and edited in Inkscape (https://inkscape.org).

2.4 Results

2.4.1 Interface used to reconstruct the GSM of *S. leeuwenhoekii*, and perform simulations

An interface called Genome Model Reconstruction Assistant (GeMRA) that helps in the reconstruction of GSM was developed. GeMRA was coded in python and uses the bioservices, COBRApy, re, and namedtuple modules, among others. The interface clumps five options: get reactions information, get metabolites, write model in COBRApy, create or load a model and perform simulations (Figure 2.2).

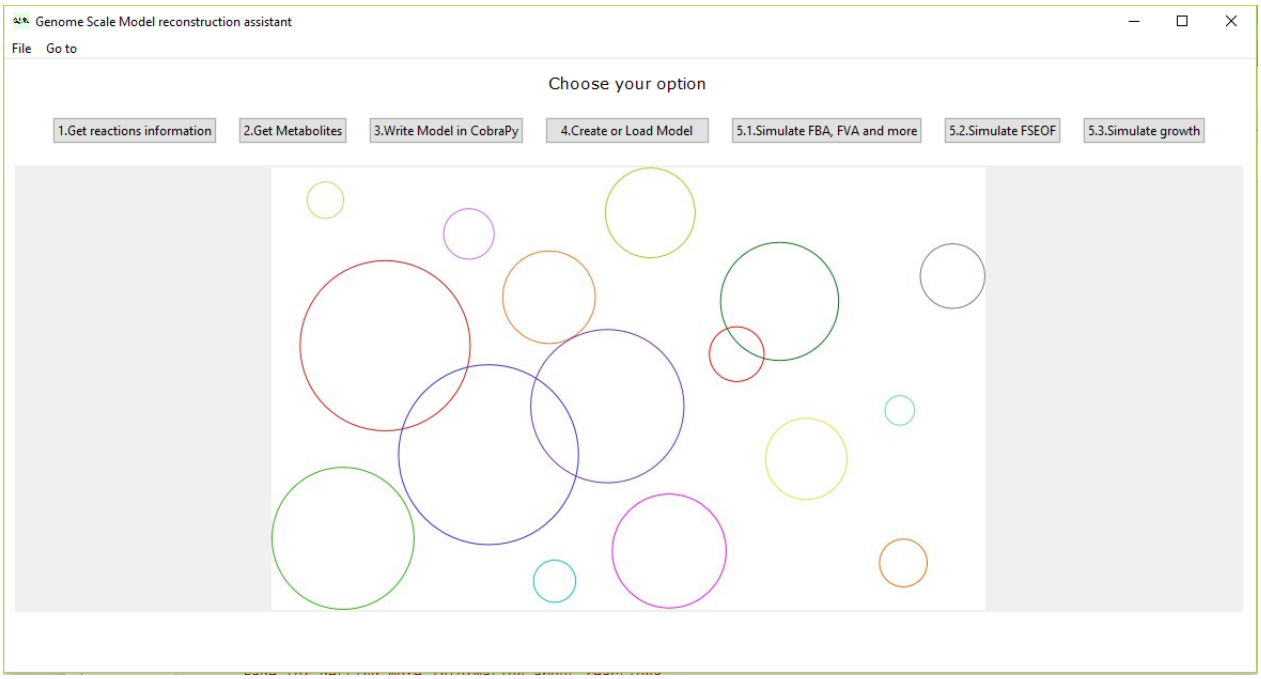


Figure 2.2: GeMRA interface developed to help in the reconstruction of GSMs and to perform simulations. The interface has five options, that includes download of information, creation of the model and perform simulations, for details see the text. For information of the code used to create the interface see Section F.3.

Get information of reactions and perform BLAST. Using the KEGG reactions numbers and the bioservices module of python, the information of GPR of *Streptomyces* strains was retrieved (code is available in the Appendix F, Section F.3.1). The information was saved in two files, one for the protein sequences and other for the reaction information. Then, a local-BLAST analysis against a *S. leeuwenhoekii* C34 protein sequences database was done using an e-value cut-off of 0.001 and at least 45% of identity (code is available in Appendix F, Section F.3.2). Later, the GPR relationship including *S. leeuwenhoekii* C34 genes was saved to a new file or appended to a previous file. Afterwards, the information obtained was curated (see Section 2.4.2). Information for 1706 reactions was obtained (without considering transport, specialised metabolites and biomass reactions), not all of these reactions were used in the model, because they were disconnected from the metabolism and/or there were gaps. In the iterative process of constructing the GSM, transport reactions and also reactions needed to explain experimental observations were included.

Download information of compounds. The information of the compounds associated with any reaction was downloaded from KEGG database using they compound numbers (code is available in Appendix F, Section F.3.3). Then, an automatic metabolites abbreviation script was used to help in the abbreviation of the compound names, taking into consideration the compounds that were already abbreviated (code is available in Appendix F, Section F.3.4). A total of 1463 compounds were included in the model (see Table F.13 in Appendix F).

Write reactions and metabolites in COBRApy format. In order to add the metabolites and reactions to the model they must be written in a format compatible with COBRApy. The files containing the compounds and reactions information were loaded and then it was written in the format required by COBRApy (code is available in Appendix F, Section F.3.5). The new information was saved and used to create the model (code is available in Appendix F, Section F.3.6).

Create or load a model. The model was created, using COBRApy module, employing the file generated in the previous step (code is available in Appendix F, Section F.3.7). If needed, the model created could be loaded. Basic data of the model and the reactions, metabolites and genes present in the model are displayed. Using COBRApy, the mass balance was checked and using the GeMRA interface it was possible to adjust protons or water if necessary (code is available in Appendix F, Section F.3.8 and F.3.9), if a further mass balance was needed it was evaluated by hand. The model was saved with all the changes.

Perform simulations. The constructed model was used to perform simulations using COBRApy and the solver Gurobi (http://www.gurobi.com/). Like FBA with different objectives function, changes of boundaries, analysis of growth in different carbon/nitrogen/phosphorous sources, FVA, gene deletion analysis and identification of gene target overexpression through FSEOF.

2.4.2 Curation of the model

The directionality of each reaction was checked and properly annotated. At the beginning the directionality was assigned using the ΔG value at pH 7 (Flamholz et al., 2011) however deeper analysis looking for bibliography information of reversibility of the reactions, showed that some of them were miss annotated, so using bibliography information available in the BRENDA database (http://www.brenda-enzymes.org/index.php), the directionality of the reactions was designated. The discrepancies found could be due to different cellular conditions (temperature, pH, ionic strength, concentration of ions and metabolites) used to predict the directionality by Flamholz et al. (2011) and the experimental conditions used in the microorganisms in BRENDA database.

Each reaction was manually curated to ensure that the GPR relationship was properly assigned. For doing so, the domains of each *S. leeuwenhoekii* C34 protein was compared to the homologous protein of the other *Streptomyces* assigned to the same reaction. From all the homologous proteins found in *S. leeuwenhoekii* C34, just 43 had different domain configurations. All of them were analysed and classified among three options: they had higher homology and small domain differences, like repeated and/or incomplete domains (Figure 2.3 A), they had different domain configurations (Figure 2.3 B) or they had lower homology but same essential domains (Figure 2.3

C). Fourteen genes were excluded from the model after this analysis. On the other hand, the protein Sle00890 had an extra domain (compared to their homologous proteins), and the extra domain was found in proteins associated to other reaction, based on that information Sle00890 was assigned as bifunctional.

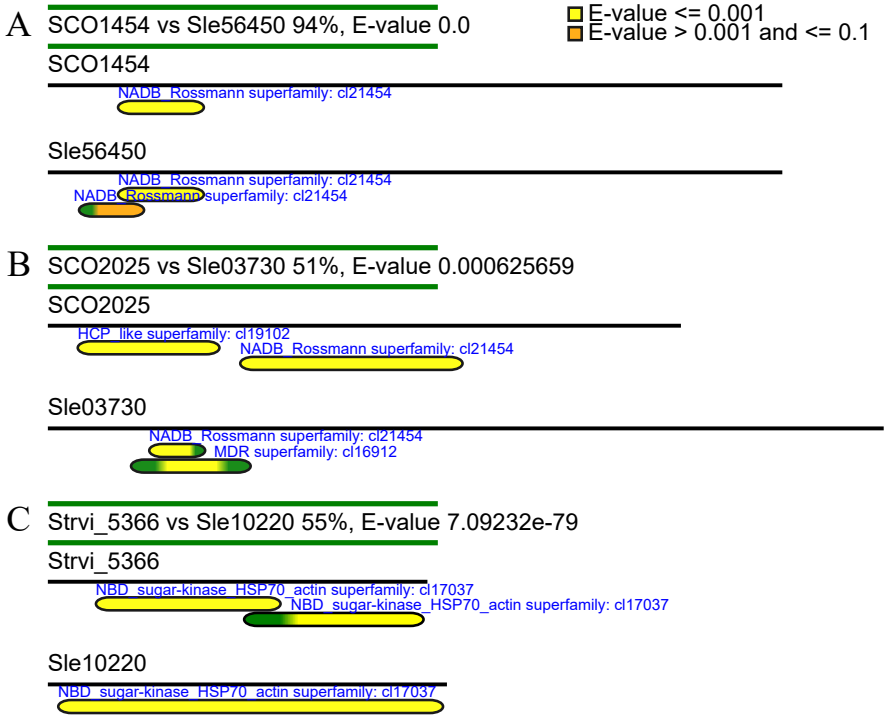


Figure 2.3: Examples of the result obtained when applied the domain comparison of proteins. The domains were slightly different **A**, the domains were completely different **B**, and the same domains were found duplicated **C**.

The information downloaded from KEGG database did not specify if the associated genes to a reaction were isoenzymes or part of protein complexes. So, each GPR relationship was analysed for proper assignment. When a reaction catalysed by protein complexes was detected, each gene association was carefully annotated, using ”AND” for complexes and ”OR” for isoenzymes.

2.4.3 Validation of the model

The default conditions for simulation of growth were set as specified in the methodology (Section 2.3.7 and 2.3.8). The simulations of growth were done using the interface GeMRA where the boundaries of the reactions were changed in order to simulate the different growth conditions.

It was checked whether the model correctly predicted growth in aerobic conditions and not growth in anaerobic conditions, by analysing the growth capabilities when the oxygen exchange reaction was blocked. The model predicted that there was no growth when the oxygen exchange reaction was blocked, and that growth depends on the uptake rate of oxygen.

The predictability of the model was evaluated and validated using data of growth/not-growth in sole carbon, nitrogen or phosphorous sources obtained in this work and Biolog data obtained by Dr. Castro in Newcastle, UK. Comparison between the prediction and experimental results are shown in Figure 2.4 and 2.5. The model correctly predicted growth/not-growth with 72 out of 89 carbon sources, 45 out of 54 nitrogen sources and 22 out of 23 phosphorous sources, corresponding to a 83.7 % of accuracy. There were five cases for which the model was not capable to predict growth contradicting the experimental information. These discrepancies could be due to missing content or inconsistencies. On the other hand there were sixteen cases where the model predicts growth and the experimental evidence showed the opposite. The differences could be due to errors, inconsistencies or missing constrains, like transcriptional regulatory constraints.

Carbon source	Abbreviation	Biolog	Prediction
N-Acetyl-D-Glucosamine	NAceDglu_e		
Succinic Acid	Succ_e		
D-Galactose	DGal_e		
L-Aspartic Acid	LAsp_e		
L-Proline	LPro_e		
D-Alanine	DAla_e		
D-Trehalose	aaTre_e		
D-Mannose	DMan_e		
D-Sorbitol	DSor_e		
Glycerol	Glyc_e		
D-Glucuronic Acid	DGluc_e		
D-Gluconic Acid	DGlucad_e		
D-Xylose	DXylo_e		
L-Lactic Acid	SLac_e		
Formic Acid	For_e		
L-Glutamic Acid	LGlu_e		
D-Glucose-6-Phosphate	DGlu6p_e		
D,L-Malic Acid	SMal_e		
D-Ribose	DRib_e		
L-Rhamnose	LRha_e		
D-Fructose	DFru_e		
Acetic Acid	Ace_e		
D-Glucose	DGlu_e		
Maltose	Mal_e		
D-Melibiose	Mel_e		
Thymidine	Thym_e		
L-Asparagine	LAspa_e		
a-keto-Glutaric Acid	dOglu_e		
a-keto-Butyric Acid	dObutte_e		
Sucrose	Suc_e		
Uridine	Uri_e		
L-Glutamine	LGlut_e		
D-Glucose-1-phosphate	DGlu1p_e		
D-Fructose-6-Phosphate	DFru1p_e		
a-Hydroxy Butyric acid	dHxybad_e		
2-Deoxy Adenosine	Dxade_e		
Adenosine	Aden_e		
Citric Acid	Cit_e		
myo-Inositol	myoIno_e		
Fumaric Acid	Fum_e		
Propionic Acid	Prote_e		
Glycolic Acid	Glyco_e		
D-Cellobiose	Cell_e		
Inosine	Ino_e		
L-Serine	LSer_e		
L-Threonine	LThr_e		
L-Alanine	LAla_e		
Acetoacetic Acid	Acet_e		
N-Acetyl-b-D-Mannosamine	NAceDman_e		
L-Malic Acid	SMal_e		
Tyramine	Tyr_e		
Pyruvic Acid	Pyr_e		
D-Galacturonic Acid	DGala_e		
N-Acetyl-Neuraminic Acid	NAceneu_e		
2-Deoxy-D-Ribose	Dxr_e		
D-Raffinose	Raf_e		
Salicin	Sal_e		
Stachyose	Sta_e		
g-Amino Butyric acid	qAmite_e		
Citraconic Acid	dMylmal_e		
4-Hydroxy Benzoic Acid	qHxybzte_e		

Carbon source	Abbreviation	Biolog	Prediction
Oxalic Acid	Oxala_e		
N-Acetyl-L-Glutamic Acid	NAceLglu_e		
L-Arginine	LArg_e		
Glycine	Gly_e		
L-Histidine	LHis_e		
L-Homoserine	LHomo_e		
Hydroxy-L-Proline	Hxy_e		
L-Isoleucine	LIsoleu_e		
L-Leucine	LLeu_e		
L-Lysine	LLys_e		
L-Methionine	LMet_e		
L-Ornithine	LOrn_e		
L-Phenylalanine	LPhenylala_e		
L-Valine	LVal_e		
Putrescine	Put_e		
L-Arabinose	LAra_e		
D-Glutarate	DGluca_e		
a-D-Lactose	Lac_e		
Glyoxylic Acid	Glyo_e		
Dextrin	Dex_e		
Arbutin	Arb_e		
D-Glucosamine	DGluco_e		
D-Glucosaminic Acid	dAmi2deoDglu_e		
b-Hydroxy Butyric Acid	R3Hxyte_e		
Oxalomalic Acid	tOxa_e		
Quinic Acid	Quin_e		
Pidolic acid	Pidoad_e		
Dihydroxy Acetone	Glyce_e		

Figure 2.4: Comparison of experimental information of growth in different carbon sources with model predictions. Growth (green); not-growth (red).

Nitrogen source	Abbreviation	Biolog	Prediction
Ammonia	NH3_e		
Nitrite	Nit_e		
Nitrate	Nitr_e		
Urea	Urea_e		
L-Alanine	LAla_e		
L-Arginine	LArg_e		
L-Asparagine	LAspa_e		
L-Aspartic Acid	LAsp_e		
L-Cysteine	LCys_e		
L-Glutamic Acid	LGlu_e		
L-Glutamine	LGlut_e		
Glycine	Gly_e		
L-Histidine	LHis_e		
L-Isoleucine	LIsoleu_e		
L-Leucine	LLeu_e		
L-Lysine	LLys_e		
L-Methionine	LMet_e		
L-Phenylalanine	LPhenylala_e		
L-Proline	LPro_e		
L-Serine	LSer_e		
L-Threonine	LThr_e		
L-Tryptophan	LTry_e		
L-Tyrosine	LTyr_e		
L-Valine	LVal_e		
D-Alanine	DAla_e		
D-Glutamic Acid	DGluta_e		
L-Citrulline	LCit_e		
L-Homoserine	LHomo_e		
L-Ornithine	LOrn_e		
N-Acetyl-L-Glutamic Acid	NAceLglu_e		
Putrescine	Put_e		
Agmatine	Agm_e		
Tyramine	Tyr_e		
Formamide	Form_e		
N-Acetyl-D-Glucosamine	NAceDglu_e		
N-Acetyl-D-Mannosamine	NAceDman_e		
Adenine	Ade_e		
Adenosine	Aden_e		
Cytidine	Cyt_e		
Cytosine	Cyto_e		
Guanine	Gua_e		
Guanosine	Guan_e		
Thymine	Thymi_e		
Thymidine	Thym_e		
Uracil	Ura_e		
Uridine	Uri_e		
Inosine	Ino_e		
Xanthine	Xan_e		
Xanthosine	Xant_e		
Uric Acid	Urate_e		
D,L-a-Amino-N-Butyric Acid	qAmite_e		
Pidolic acid	Pidoad_e		
D-Glucosamine	DGluco_e		
Allantoin	SAll_e		

Phosphorous source	Abbreviation	Biolog	Prediction
Phosphate	pi_e		
Pyrophosphate	ppi_e		
Triphosphate	Trip_e		
Adenosine 5'monophosphate	AMP_e		
Carbamyl Phosphate	Carbp_e		
D-2-Phospho-Glyceric Acid	dPDgly_e		
D-3-Phospho-Glyceric acid	tPDgly_e		
Guanosine 5'-monophosphate	GMP_e		
Guanosine 3',5' cyclic monophosphate	t5CyclGMP_e		
Phosphoenol Pyruvate	Pennolpyr_e		
Phospho-Glycolic Acid	dPgl_e		
D-Glucose-1-phosphate	DGlu1p_e		
D-Glucose-6-Phosphate	DGlu6p_e		
D-Glucosamine-6-Phosphate	DGluc6p_e		
Cytidine-5'-monophosphate	CMP_e		
D-Mannose-1-Phosphate	DMann1p_e		
D-Mannose-6-Phosphate	DMann6p_e		
O-Phospho-L-Serine	OPLser_e		
O-Phospho-L-Threonine	LThreO3p_e		
Uridine 5'-monophosphate	UMP_e		
Phosphoryl Choline	Cholp_e		
O-Phosphoryl-Ethanolamine	Ethap_e		
Thymidine 5'-monophosphate	dTMP_e		

Figure 2.5: Comparison of experimental information of growth in different nitrogen and phosphorous sources with model predictions. Growth (green); not-growth (red).

For allowing the model to use several carbon/nitrogen sources, it was necessary to include gap reactions. For example, the experimental evidence showed that *S. leeuwenhoekii* C34 was capable of growing using L-Lysine as carbon/nitrogen source, however the reactions required to allow L-Lysine consumption and connection to the metabolic network did not have the *S. leeuwenhoekii*

C34 genes associated. Further studies are required in order to identify the genes involved.

The metabolism of D-melibiose, D-raffinose and satchyose goes through the production of D-galactose. The model predicts that *S. leeuwenhoekii* C34 is able to grow using either of the mentioned carbon sources, however the experimental information said that there is no growth with D-melibiose, D-raffinose and satchyose, while there is growth with D-galactose. The conversion of D-melibiose, D-raffinose or stachyose to D-galactose is catalyzed by the genes *sle10440*, *sle08850* or *sle63600*. Additional analyses are required in order to see if these reactions are subject to transcriptional constraints.

Similar cases occur for D-alanine, formic acid, D-glucose 6-phosphate, sucrose, uridine, glycolate, L-homoserine, L-leucine, L-ornithine, D-glutamate, cytidine and cytosine, all the genes needed to use these carbon/nitrogen sources are present in *i*VR1007 and consequently the model predicts growth, however the experimental evidence showed that *S. leeuwenhoekii* was not able to use those sole carbon/nitrogen sources. This suggests that there is a lack of either regulatory constraints.

The model predicts growth with L-lactic acid as carbon source in contrast with experimental evidence. The metabolism of this compound in the model includes gap reactions that were added to complete the fructose and mannose metabolism. If these reactions were eliminated from the model, *i*VR1007 predicts not-growth with L-rhamnose contradicting the experimental information. Similarly is the case of formamide, where if the gap reaction is deleted the model wrongly predicts growth with other sources.

The experimental information showed that D-glucosamine could be used as sole carbon source but not as nitrogen source. In contrast, the model predicts that it is possible to use it as either carbon or nitrogen source. Further studies are required in order to understand why *S. leeuwenhoekii* is not capable of using this substrate as nitrogen source.

On the other hand, the capability to grow in complex media (Table F.12) was also evaluated. The model predicted a higher growth rate in complex media compared to growth on sole carbon sources.

2.4.4 The model

A model consistent and validated was obtained and named *i*VR1007. The model *i*VR1007 has 1722 reactions, 1463 metabolites and 1007 genes. There are 425 reactions that do not have an associated gene but are needed for connectivity of the model or for biomass production (Table 2.1). The distribution of *i*VR1007 reactions is shown in Figure 2.6.

GAM requirements were assumed to be 40 mmol ATP gDW^{-1} h^{-1} as observed for *S. coelicolor* (Borodina et al., 2005). This term was added as ATP hydrolysis to the biomass artificial reaction. While, NGAM requirements are imposed by setting the lower bound of the ATPM reaction to 3

mmol ATP gDW^{-1} h^{-1} as stated for *S. coelicolor* (Borodina et al., 2005).

Table 2.1: Statistics of *i*VR1007.

	Number
Total reactions	1722
Metabolic conversions	1483
Transport reactions	239
Exchange reactions	186
Reactions with ORF assigned	1297
Reactions without ORF assigned (GAPs)	425
Percentage of reactions with ORF	75.2 %
Transport reactions with ORF	108
Percentage of transport reactions with ORF	45.2 %
Metabolic conversions with ORF	1189
Percentage of metabolic conversions with ORF	80.0 %

The pathways needed for the use of different carbon/nitrogen and phosphorous sources were included in the model, specially if there was experimental evidence of growth in those sources. For some cases, it was necessary to include gaps reactions, such as those for allowing the degradation of D-galacturonate and L-rhamnose.

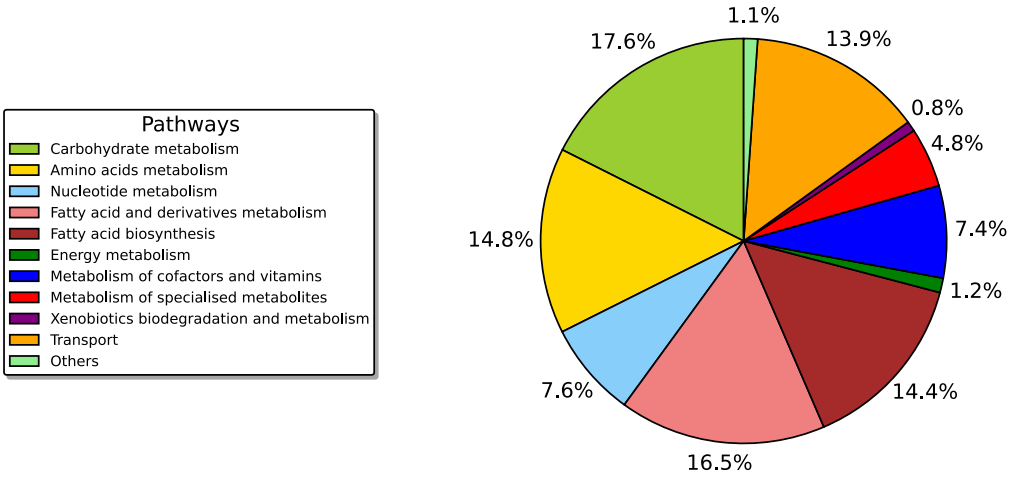


Figure 2.6: Distribution of reactions of *i*VR1007 in each metabolism category.

2.4.4.1 Fatty acid biosynthesis

The fatty acid composition of *S. leeuwenhoekii* C34 has been described by (Busarakam, 2014) (see supplementary Table F.5). According to the bibliographic information, it was necessary to include biosynthesis of branched-chain fatty acids (BCFA) as well as straight-chain fatty acids (SCFA). BCFA are biosynthesized from branched-chain amino acids, 2-methylpropanoyl-CoA, 3-methylbutanoyl-CoA and (S)-2-methylbutanoyl-CoA, and uses malonyl-CoA as extender units, to produce *iso*- (odd numbered chain), *iso*- (even numbered chain) and *anteiso*- (even numbered chain) fatty acids, respectively. Also, *i*VR1007 includes the biosynthesis of unsaturated fatty acids.

Malonyl-CoA is produced from acetyl-CoA in the reactions rx0404 and rx0405 (in KEGG: r04386 and r04385, respectively)

All the reactions of fatty acid biosynthesis were added stepwise instead of being lumped in one reaction. Because of that there are several gaps reactions in the fatty acid biosynthesis and metabolism (Figure 2.7)

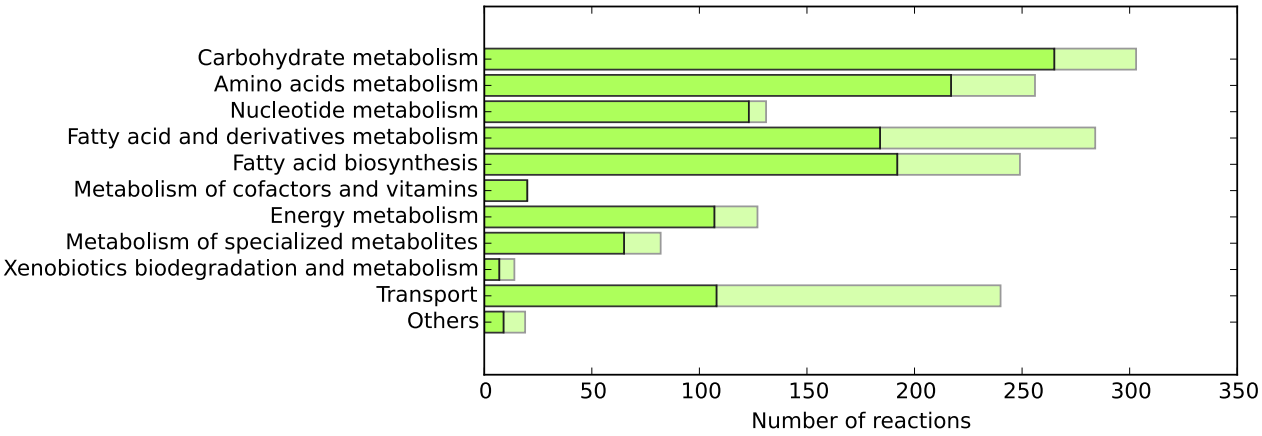


Figure 2.7: Reactions and gaps distribution of *i*VR1007 in the metabolic pathways.

2.4.4.2 Specialised metabolites pathways

Specialised metabolites like polyketides or non-ribosomal peptides are built with precursors synthesised during the primary metabolism. PKSs use acyl-CoA molecules as building blocks such as acetyl-CoA, malonyl-CoA or (S)-methylmalonyl-CoA. On the other hand, NRPSs use amino acids and derivatives as precursors and extender units. The Shikimate pathway is important for generation of amino acids and derivatives that are used during the primary metabolism and also in the biosynthesis of specialised metabolites. Reactions needed for the biosynthesis of chaxamycins, chaxalactins, desferrioxamines, ectoine and hybrid PK-NP were included in the model.

Chaxamycins. Chaxamycins are novel polyketide antibiotics with activity against *Staph. aureus* and also have inhibitory effect over the human Hsp90 protein (antitumor activity) (Rateb et al., 2011a). The proposed metabolic pathway of chaxamycins biosynthesis (Castro et al., 2015) is based in the rifamycin biosynthesis. Biosynthesis reactions of chaxamycins were lumped according to the PKS genes (Figure 2.8). A total of 21 reactions that were exclusive, were included in the model in order to allow production of chaxamycin A and B. The key precursors needed for the biosynthesis of chaxamycins are: 3-Amino-5-hydroxybenzoate (AHBA), malonyl-CoA and (S)-Methylmalonyl-CoA.

Chaxalactins. Chaxalactins are novel polyketide antibiotics with activity against *Staph. aureus* (Rateb et al., 2011b). There are three types of chaxalactins but since all of them use the same precursors it has been proposed that chaxalactin A is synthesised first; just chaxalactin A biosynthesis reactions were added to the model. The reactions needed for chaxalactin A biosynthesis was inferred from the putative biosynthetic pathway proposed (Castro, 2015). There are five PKS biosynthesis genes and each of them was annotated as a reaction (Figure 2.9).

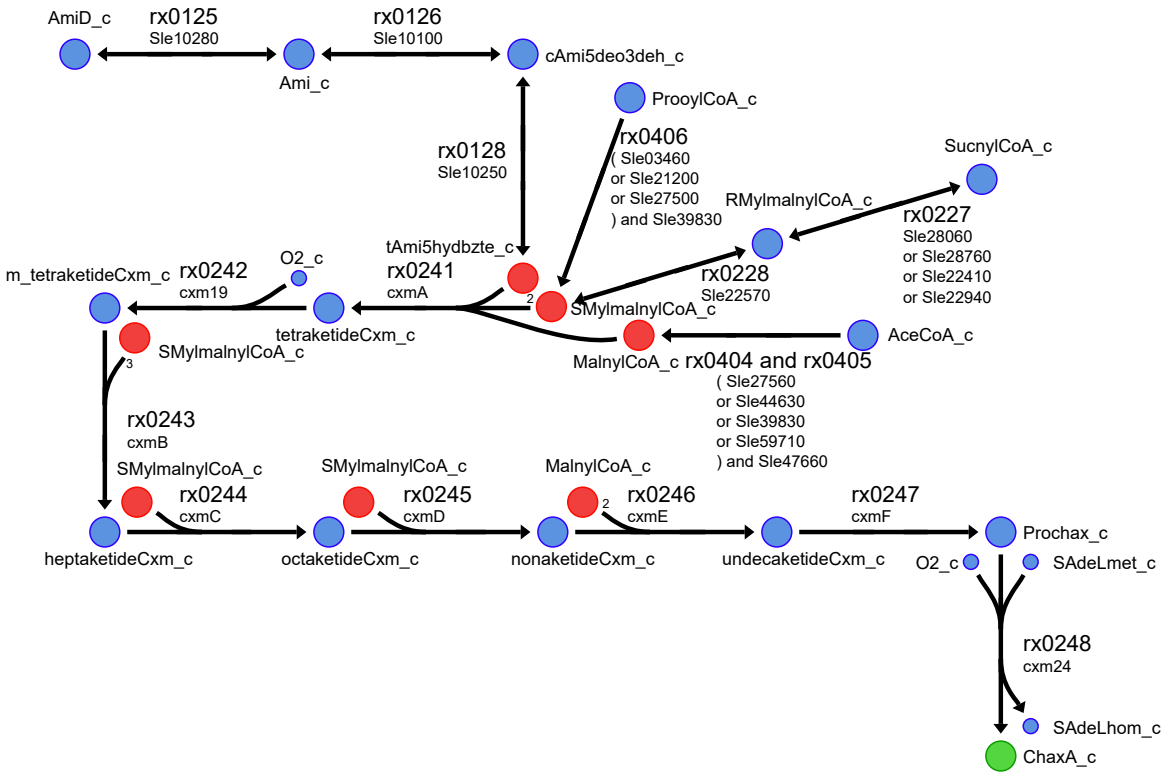


Figure 2.8: Chaxamycin A biosynthesis pathway. Green: specialised metabolites product, red: main used precursors, blue: intermediates metabolites. For description of the abbreviations go to Table F.13.

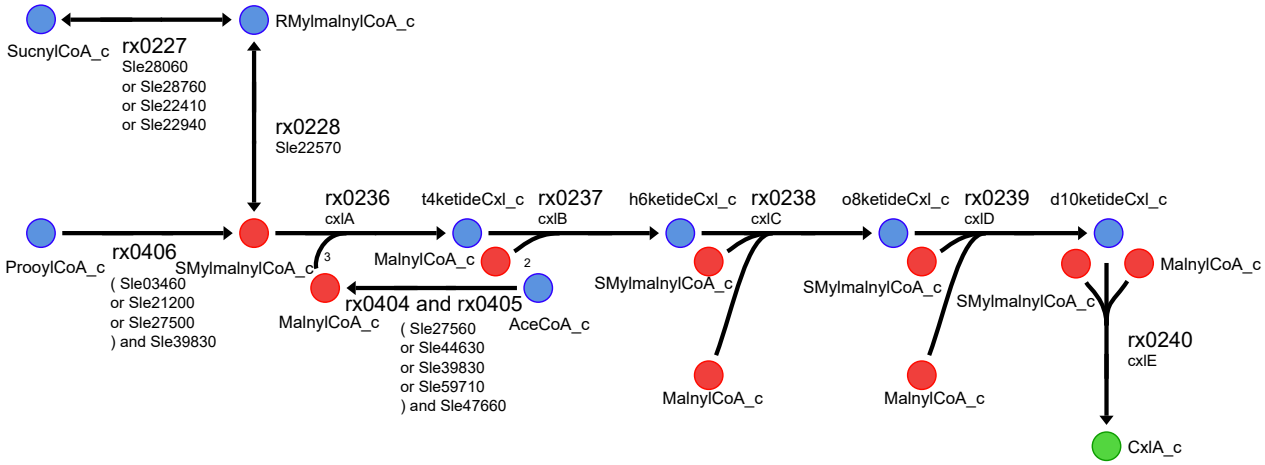


Figure 2.9: Chaxalactin A biosynthesis pathway. Green: specialised metabolites product, red: main used precursors, blue: intermediates metabolites. For description of the abbreviations, go to Table F.13.

Desferrioxamines. Desferrioxamines are siderophores with high affinity to iron. They are produced when the concentration of iron is low. Desferrioxamines are released by the cells, they form an iron complex and then the complex is transported into the cell to recover the iron. The BGC of desferrioxamines has been studied for *S. coelicolor* (Barona-Gómez et al., 2004). In the same work the biosynthesis pathway of desferrioxamine E was described, complementing the previous information (Schupp et al., 1988; Günter et al., 1993). The biosynthesis pathway is shown in Figure 2.10.

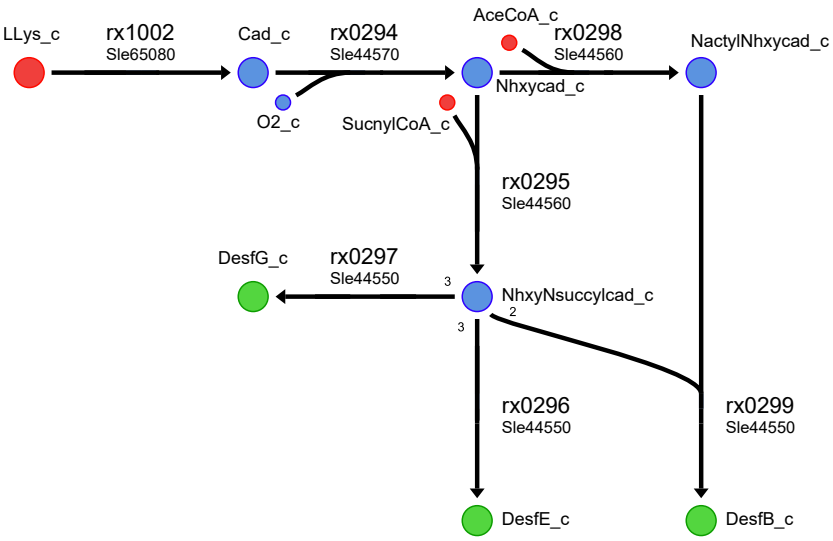


Figure 2.10: Desferrioxamine B, E and G biosynthesis pathway. Green: specialised metabolites product, red: main used precursors, blue: intermediates metabolites. For description of the abbreviations, go to Table F.13.

Ectoine. Ectoine is a compatible solute that has a protective role within the cell. It is produced in higher amounts when the cell is under salinity or heat stress. The biosynthesis pathway of ectoine has been previously described (Peters et al., 1990; Bursy et al., 2008). Ectoine is synthesized by enzymes encoded in a highly conserved BGC: *ectABC*. Homologous genes of the ectoine BGC have been identified in the genome of *S. leeuwenhoekii* C34. The biosynthesis pathway of ectoine is shown in Figure 2.11.

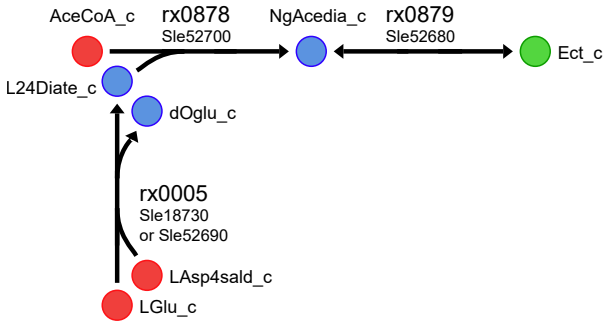


Figure 2.11: Ectoine biosynthesis pathway. Green: specialised metabolites product, red: main used precursors, blue: intermediates metabolites. For description of the abbreviations, go to Table F.13.

Hybrid PK-NP. In order to study the precursor supply for the production of the hybrid PK-NP, hypothetical reactions based on the putative biosynthetic pathway of the hybrid *trans*-AT PKS/NRPS BGC (described in Chapter 1) were included in the model. For simplicity, three reactions were added. The first one incorporates the starter unit (L-threonine), the second incorporate the extensor unit of the NRPS (L-cysteine) and the extensor units of the PKS (malonyl-CoA). The third reaction involved the release and cyclisation of the compound (Figure 2.12).

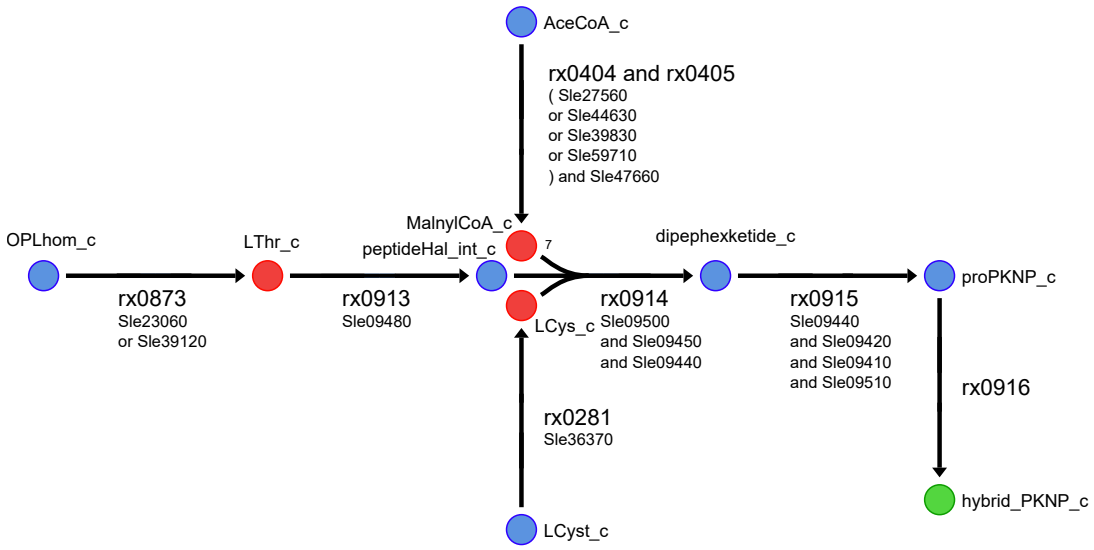


Figure 2.12: Hybrid transAT PKS/NRPS biosynthesis pathway. Green: specialised metabolites product, red: main used precursors, blue: intermediates metabolites. For description of the abbreviations go to Table F.13.

2.4.5 Essential gene analysis

Simulation of growth in complex medium was done for analysing the effect of gene deletion under these conditions, in order to identify the genes that are essential to growth. Each gene in the model was deleted and the biomass production rate was optimized. This allowed obtaining a set of reactions that were essential for growth under the conditions studied. In Table 2.2 there is a summary of all the gene deletions that change the growth of *S. leeuwenhoekii* C34. There are 76 genes that are essential for growth in complex media (Table F.11), they are involved in reactions of several metabolic pathways, amongst the main metabolic pathways there is the purine and pyrimidine metabolism, fatty acid biosynthesis and Glycerophospholipid metabolism.

Table 2.2: Essential gene analysis.

Effect of gene deletion	Number of genes
No growth	76
Reduce growth to 5 -25 %	9
Reduce growth to 25 - 50 %	30
Reduce growth to 50 - 75 %	16
Reduce growth to 75 - 90 %	7
Reduce growth to 90 - 99.99 %	43
Growth not affected	826

2.4.6 Gene knockout analysis

Non-intuitive genes knockout that could improve specialised metabolites production were identified using the model and the methodology described in Section 2.3.9.

In a first step, the procedure allowed to obtain sets of gene deletions that increment the produc-

tion of chaxalactin A and hybrid PK-NP. For increasing chaxamycin A production no sinlge single gene knockout was found.

The gene *sle03600* is associated to the reaction rx0268 (KEGG No. r01776) in the cysteine and methionine metabolism. Deletion of this gene produced more availability of acetyl-CoA that could be converted to malonyl-CoA and increase chaxalactin A production to a fraction of 0.6 of the theoretical maximum yield. The deletion of *sle03600* was used as starting point to predict further improvement of chaxalactin A production. It was found that an additional deletion of the genes *sle21250*, *sle22000* and *sle47020*, associated to the reactions rx0022 (KEGG No. r01213) predicted an increase in chaxalactin A production to 0.67 of the maximum yield (Figure 2.13).

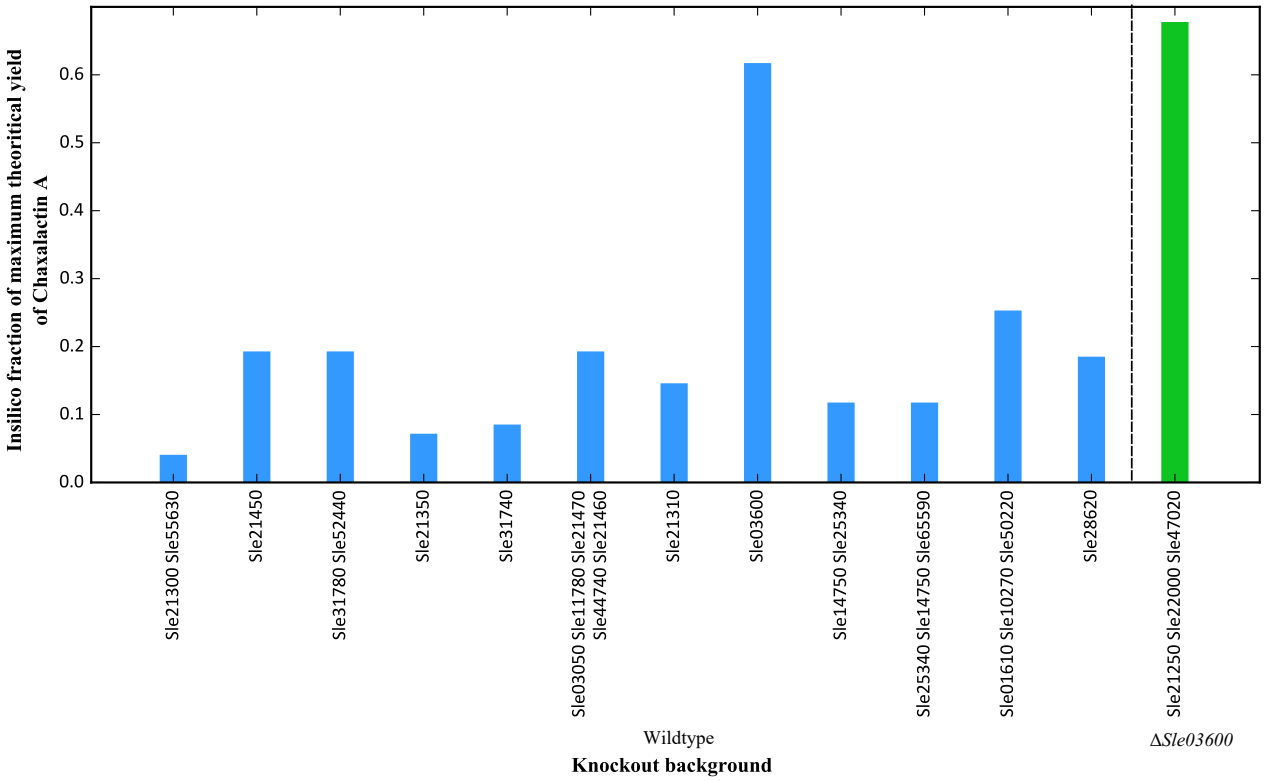


Figure 2.13: Gene knockout search for increased chaxalactin A production in the wildtype strain (blue) and Δ*sle03600* (green).

From the set of gene knockouts detected to increase hybrid PK-NP production, the deletion of gene *sle17360* predicted the highest increment (0.1 of the theoretical maximum). Later, using Δ*sle17360* as genetic background, it was predicted that the deletion of *sle39090* improved hybrid PK-NP production to over 0.6 of the maximum (Figure 2.14).

It was also possible to detect gene knockouts using Δ*sle03600* as background. The gene deletion that predicted the highest increments in hybrid PK-NP production were: *sle47670* associated to the reaction rx1582 (KEGG No. r03383) and the genes *sle44610*, *sle44660*, *sle54280* and *sle61230* associated to the reactions rx0309, rx0310 and rx0314 (KEGG No. r02661, r03172, r04095, respectively) (Figure 2.15).

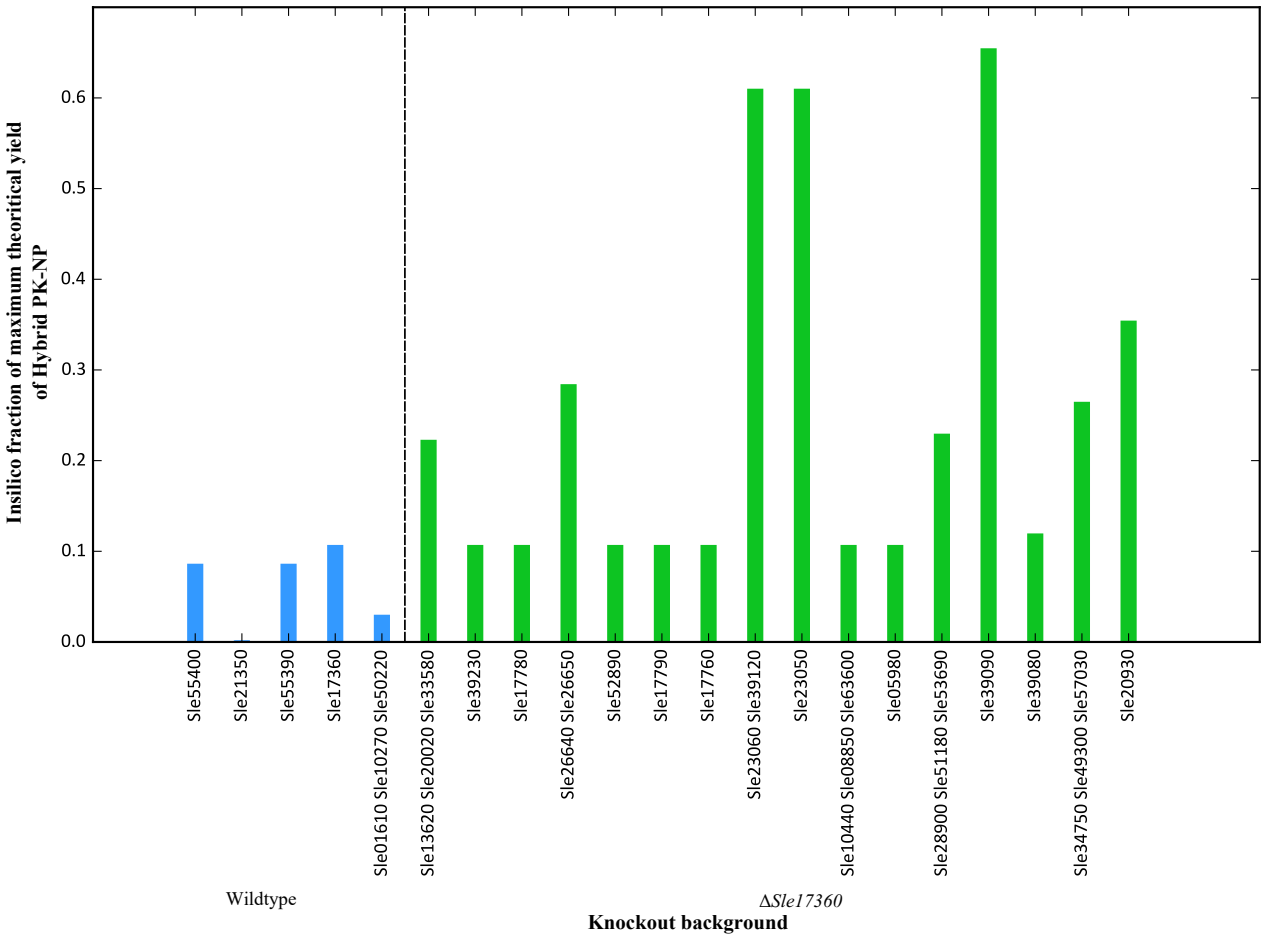
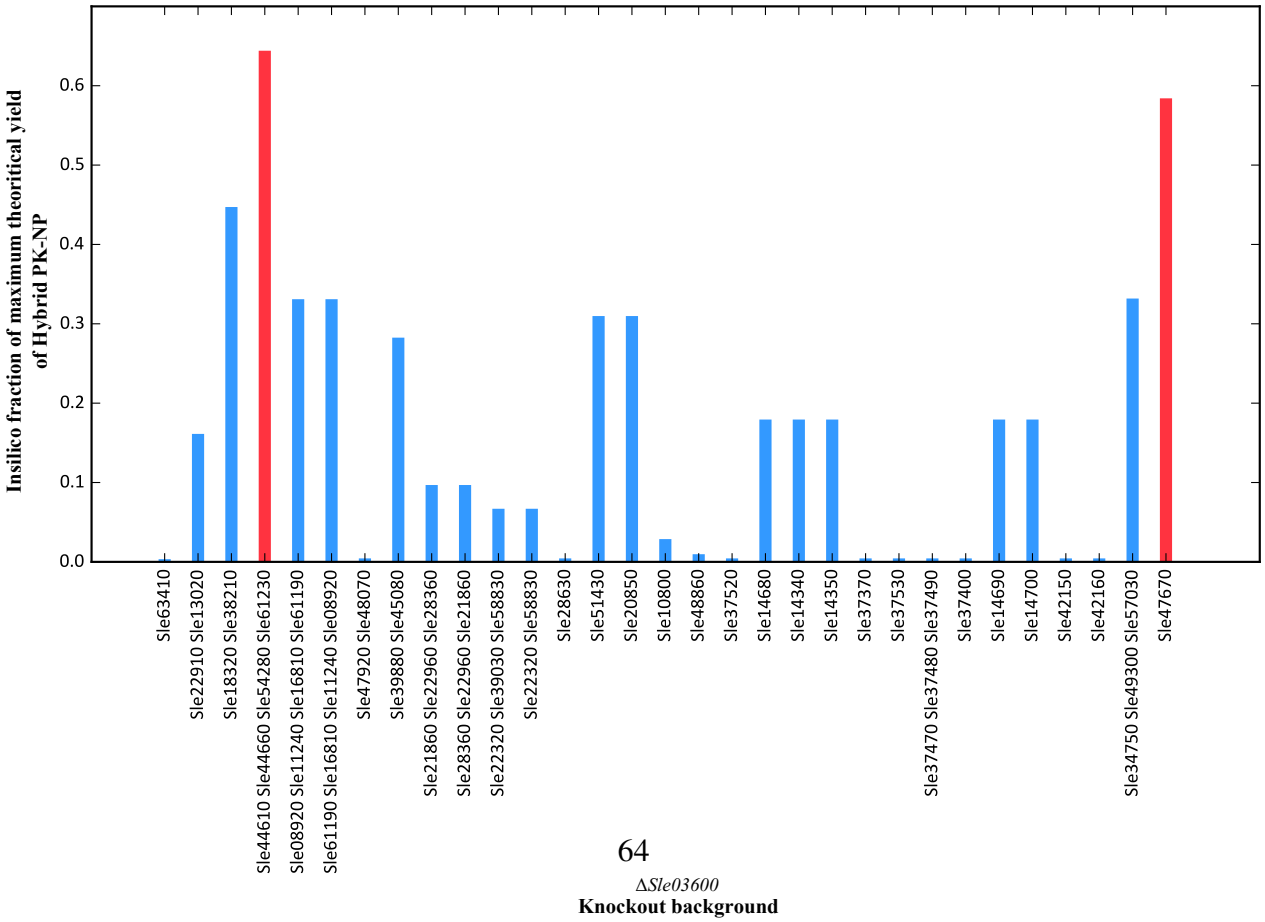


Figure 2.14: Gene knockout search for increased hybrid PK-NP production in the wildtype strain (blue) and Δ*sle17360* (green) background.



The chaxamycin A production was predicted to be increased in the presence of at least double gene deletions. The deletion of *sle03600* and *sle39090* produced an increase to 0.08 of the theoretical maximum. Further deletion of *sle29000*, *sle29010* and *sle56410* associated to the reaction rx1231 (KEGG No. r02661, r01130) predicted an improvement of chaxamycin A production to a fraction of 0.44 of the theoretical maximum (Figure 2.16).

A summary of the predicted gene knockouts that enhance the production of specialised metabolites and their associated reactions is shown in Table 2.3.

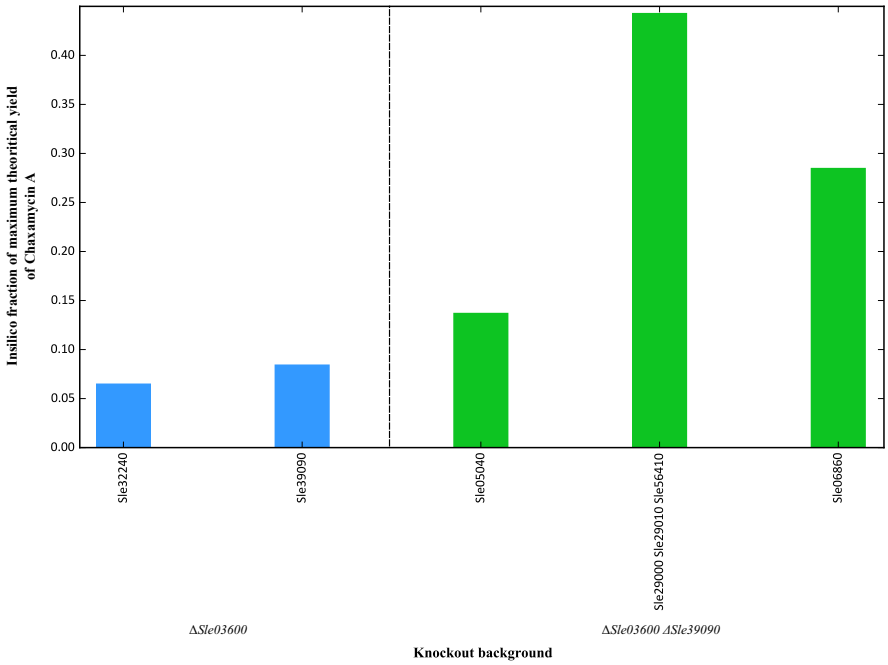


Figure 2.16: Gene knockout search for increasing chaxamycin A production in Δ*sle03600* (blue) and Δ*sle39090* (green) background.

Table 2.3: Predicted genes knockout targets for increasing chaxamycin A, chaxalactin A, and hybrid PK-NP production.

Predicted effect of knockout	**Gene(s)**	**No.**	**Reaction**
Increase chaxalactin A production	*sle03600*	rx0268	AceCoA_c + LHomo_c ↔ CoA_c + OAceL-hom_c
Increase chaxalactin A production in a Δ*sle03600* background	*sle21250*, *sle22000*, *sle47020*	rx0022	AceCoA_c + tMyl2obutad_c + H2O_c → aIsopro_c + CoA_c + H_c
Increase hybrid PK-NP production	*sle17360*	rx0085	Carbp_c + LOrn_c ↔ pi_c + LCit_c + 2 H_c
Increase chaxamycin A production in a Δ*sle03600* background. Increase hybrid PK-NP production in a Δ*sle17360* background	*sle39090*	rx1350	UDPglu_c + aDGl6p_c → UDP_c + aa-Treh6p_c
Increase chaxamycin A production in a Δ*sle03600* and Δ*sle39090* background	*sle29000*, *sle29010*, *sle56410*	rx1231	IMP_c + NAD_c + H2O_c → Xant5p_c + NADH_c + H_c
Increase hybrid PK-NP production in a Δ*sle03600* background	*sle47670*	rx1582	H_c + Mylmal_c + CoA_c + ATP_c → RMyl-malnylCoA_c + AMP_c + ppi_c

Continuation of Table 2.3

Predicted effect of knockout	Gene(s)	No.	Reaction
Increase hybrid PK-NP production in a Δ*sle03600* background	*sle44610*, *sle44660*, *sle54280*, *sle61230*	rx0309	dMylprooylCoA_c + Acc_c → dMylpro2enooylCoA_c + Reduacce_c
		rx0310	S2MylbutoylCoA_c + Acc_c → dMylbut2enooylCoA_c + Reduacce_c
		rx0314	tMylbutoylCoA_c + FAD_c → tMylcronylCoA_c + FADH2_c

2.4.7 Identification of targets for overexpression

FSEOF was applied to identify reactions that had an incremented flux when a specific specialised metabolites production was enforced and the production of biomass was optimized under growth in complex media (Table F.12), this procedure allowed to obtain overexpression gene targets that would enhance the production of chaxamycins, chaxalactins and hybrid PK-NP (Table 2.6 and Table 2.5).

The numbers of overexpression reaction targets found that are directly and not directly related to their biosynthesis are shown in Table 2.4. Fifty-four reactions that had increasing fluxes while the production of chaxamycin A was enforced were identified. Nineteen of them are directly involved in the production of the precursor AHBA and in the production and transport of chaxamycin A, while 35 reactions are not. When FSEOF was applied to study targets of overexpression for chaxalactins enhancement, a total of 31 reactions were detected as possible targets. Thirty-two gene targets for overexpression were found for the hybrid PK-NP and 27 were not directly associated to their biosynthetic pathway.

Table 2.4: No. of overexpression targets found for each specialised metabolites.

Specialised metabolite	No. of overexpression targets		Total
	not-directly related	directly related	
Chaxamycin	35	19	54
Chaxalactin	25	6	31
Hybrid_PK-NP	27	5	32

Of the not directly related reactions several of them were found for more than one of the specialised metabolites (Figure 2.17). The identification of common overexpression targets are probably due the use of the same extensor unit (malonyl-CoA).

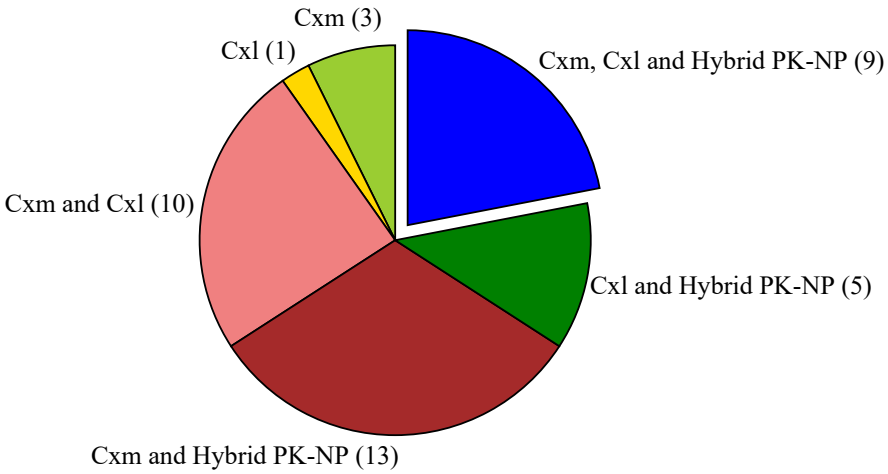


Figure 2.17: Number of overexpression gene targets non-related to the biosynthesis pathways of specialised metabolites that are shared or unique for increasing chaxamycin A (Cxm), chaxalactin A (Cxl) and hybrid PK-NP biosynthesis. For details of the reactions and genes see Table 2.5.

Table 2.5: Overexpression targets not directly related to the production of the compound of interest[1].

Compound[2]	Target	Reaction	Gene(s)
Cxm	rx0264	H_c + LMet_c + SAdeLhom_c ↔ LHom_c + SAdeLmet_c	*sle15450*
Cxm	rx0283	LHom_c + cMylthfol_c ↔ LMet_c + Thy_c	*sle54600*
Cxm	rx1367	DGlu1p_c + UTP_c ↔ UDPglu_c + ppi_c	*sle41020*
Cxl	rx1226	aDRibo1p_c ↔ DRibo5p_c	*sle66460*
Cxm and Cxl	rx0227	SucnylCoA_c ↔ RMylmalnylCoA_c	*sle28060* or *sle28760* or *sle22410* or *sle22940*
Cxm and Cxl	rx0228	RMylmalnylCoA_c ↔ SMylmalnylCoA_c	*sle22570*
Cxm and Cxl	rx1257	H2O_c + NAD_c + Xan_c → H_c + NADH_c + Urate_c	(*sle04630* and *sle04640* and *sle04650*) or (*sle59590* and *sle59600* and *sle59610*)
Cxm and Cxl	rx1258	H2O_c + O2_c + Urate_c ↔ H202_c + cHxyisoura_c	*sle14680*
Cxm and Cxl	rx1262	Allte_c + H2O_c ↔ Ure_c + Urea_c	*sle14340*
Cxm and Cxl	rx1263	H2O_c + SAll_c → Allte_c + H_c	*sle14350*
Cxm and Cxl	rx1276	H2O_c + cHxyisoura_c ↔ H_c + cHxy2o4ure25dih1Himi5car_c	*sle14690*
Cxm and Cxl	rx1277	H_c + cHxy2o4ure25dih1Himi5car_c → CO2_c + SAll_c	*sle14700*
Cxm and Cxl	rx1551	LTry_c + O2_c → LFry_c	*sle33790*
Cxm and Cxl	rx1628	Ure_c ↔ Glyo_c + Urea_c	
Cxm and Hybrid_PK-NP	rx1210	AMP_c + ATP_c ↔ 2,0 ADP_c	*sle29430*
Cxm and Hybrid_PK-NP	rx1213	ATP_c + Aden_c → ADP_c + AMP_c	*sle49810*
Cxm and Hybrid_PK-NP	rx1225	AMP_c + cPaDribo1dp_c ↔ ATP_c + DRibo5p_c	*sle05540* or *sle41570*
Cxm and Hybrid_PK-NP	rx1240	ADP_c + dADP_c ↔ ATP_c + dAMP_c	*sle29430*
Cxm and Hybrid_PK-NP	rx1242	Ade_c + aDRibo1p_c ↔ Aden_c + H_c + pi_c	*sle27600*
Cxm and Hybrid_PK-NP	rx1252	ADP_c + Thi_c → H2O_c + Thiodisu_c + dADP_c	*sle18670* or (*sle25130* and *sle25140*)
Cxm and Hybrid_PK-NP	rx1254	H2O_c + dAMP_c → Dxade_c + H_c + pi_c	*sle38050*
Cxm and Hybrid_PK-NP	rx1265	Dxade_c + H_c + pi_c ↔ Ade_c + dDxDribo1p_c	*sle27600*
Cxm and Hybrid_PK-NP	rx1280	ADP_c + UDP_c ↔ ATP_c + UMP_c	*sle20370* or *sle53710*
Cxm and Hybrid_PK-NP	rx1291	UMP_c + ppi_c ↔ Ura_c + cPaDribo1dp_c	*sle37190* or *sle56210*
Cxm and Hybrid_PK-NP	rx1312	ATP_c + dUMP_c ↔ ADP_c + dUDP_c	*sle32590*
Cxm and Hybrid_PK-NP	rx1313	ATP_c + Dxu_c → ADP_c + dUMP_c	*sle18340*
Cxm and Hybrid_PK-NP	rx1319	Ura_c + dDxDribo1p_c ↔ Dxu_c + H_c + pi_c	*sle27600* or *sle27840*

Continuation of Table 2.5

Compound[2]	Target	Reaction	Gene(s)
Cxl and Hybrid_PK-NP	rx0291	H_c + Pyr_c + Thio_c → Mer_c + Sul_c	*sle18320* or *sle38210*
Cxl and Hybrid_PK-NP	rx1094	DRibo5p_c + DXylu5p_c ↔ DGlyc3p_c + Sedo7p_c	*sle11600* or *sle12830* or *sle52070*
Cxl and Hybrid_PK-NP	rx1098	DRibo5p_c ↔ DRibu5p_c	*sle46110* or *sle58730*
Cxl and Hybrid_PK-NP	rx1103	DRibu5p_c ↔ DXylu5p_c	*sle56380*
Cxl and Hybrid_PK-NP	rx1711	LGlu_c + Mer_c ↔ LCys_c + dOglu_c	*sle12550*
Cxm, cxl and Hybrid_PK-NP	rx0288	CoA_c + OAceLser_c ↔ AceCoA_c + LSer_c	
Cxm, cxl and Hybrid_PK-NP	rx0404	AceCoA_c + Carbcarbcarrprot_c → Holca_c + MalnylCoA_c	*sle47660*
Cxm, cxl and Hybrid_PK-NP	rx0405	ATP_c + HCO3_c + Holca_c → ADP_c + Carbcarbcarrprot_c + H_c + pi_c	*sle59710* or *sle44630* or *sle39830* or *sle27560*
Cxm, cxl and Hybrid_PK-NP	rx1057	CO2_c + H2O_c → HCO3_c + H_c	*sle16220* or *sle32790* or *sle50480*
Cxm, cxl and Hybrid_PK-NP	rx1090	DGlyc3p_c + Sedo7p_c ↔ DEryt4p_c + bDFr6p_c	*sle11610* or *sle52060*
Cxm, cxl and Hybrid_PK-NP	rx1095	DEryt4p_c + DXylu5p_c ↔ DGlyc3p_c + bDFr6p_c	*sle11600* or *sle12830* or *sle52070*
Cxm, cxl and Hybrid_PK-NP	rx1708	Glu_c + LCys_c + NADP_c ↔ NADPH_c + SGlunylLcys_c	*sle36080* or *sle65870*
Cxm, cxl and Hybrid_PK-NP	rx1709	SGlunylLcys_c + Sulf_c ↔ Glu_c + SSulLcys_c	
Cxm, cxl and Hybrid_PK-NP	rx1710	Ace_c + SSulLcys_c ↔ OAceLser_c + Thio_c	*sle43610*

[1]The gene targets are shown sorted by the compound predicted to be enhanced (colours).
[2]Chaxamycins (cxm); chaxalactins (cxl).

Table 2.6: Overexpression targets directly related to the production of the compound of interest[1].

Pathway[2]	Target	Reaction	Gene(s)
Cxm	rx0119	LGlu_c + UDP3ket_c ↔ H_c + UDPkan_c + dOglu_c	*sle10250*
Cxm	rx0120	UDPkan_c ↔ Kan_c + UDP_c	*sle10230*
Cxm	rx0121	ATP_c + H_c + Kan_c ↔ ADP_c + Kano6p_c	*sle10220*
Cxm	rx0122	Kano6p_c ↔ Amif6p_c	*sle05960*
Cxm	rx0123	Amif6p_c + DRibo5p_c ↔ Imin4p_c + Sedo7p_c	*sle09700* or *sle52070* or *sle11600*
Cxm	rx0124	H2O_c + Imin4p_c + Pennolpyr_c ↔ AmiD_c + pi_c	*sle10270*
Cxm	rx0125	AmiD_c ↔ Ami_c + 2.0 H_c + pi_c	*sle10280*
Cxm	rx0126	Ami_c ↔ H2O_c + cAmi5deo3deh_c	*sle10100*
Cxm	rx0127	NAD_c + UDPglu_c ↔ NADH_c + UDP3ket_c	*sle10240*
Cxm	rx0128	cAmi5deo3deh_c ↔ H2O_c + tAmi5hydbzte_c	*sle10250*
Cxm	rx0241	ATP_c + 2.0 H_c + MalnylCoA_c + NADPH_c + 2.0 SMylmalnylCoA_c + tAmi5hydbzte_c → AMP_c + 3.0 CO2_c + 3.0 CoA_c + NADP_c + ppi_c + tetraketideCxm_c	*cxmA*
Cxm	rx0242	O2_c + tetraketideCxm_c → H2O_c + 2.0 H_c + m_tetraketideCxm_c	*cxm19*
Cxm	rx0243	6.0 H_c + 3.0 NADPH_c + 3.0 SMylmalnylCoA_c + m_tetraketideCxm_c → 3.0 CO2_c + 3.0 CoA_c + H2O_c + 3.0 NADP_c + heptaketideCxm_c	*cxmB*
Cxm	rx0244	2.0 H_c + NADPH_c + SMylmalnylCoA_c + heptaketideCxm_c → CO2_c + CoA_c + NADP_c + octaketideCxm_c	*cxmC*
Cxm	rx0245	2.0 H_c + NADPH_c + SMylmalnylCoA_c + octaketideCxm_c → CO2_c + CoA_c + NADP_c + nonaketideCxm_c	*cxmD*
Cxm	rx0246	4.0 H_c + 2.0 MalnylCoA_c + 2.0 NADPH_c + nonaketideCxm_c → 2.0 CO2_c + 2.0 CoA_c + 2.0 H2O_c + 2.0 NADP_c + undecaketideCxm_c	*cxmE*
Cxm	rx0247	undecaketideCxm_c → H_c + Prochax_c	*cxmF*
Cxm	rx0248	O2_c + Prochax_c + SAdeLmet_c → ChaxA_c + H202_c + H_c + SAdeLhom_c	*cxm24*
Cxm	rx1515	ChaxA_c → ChaxA_e	
Cxl	rx0236	7.0 H_c + 3.0 MalnylCoA_c + 3.0 NADPH_c + SMylmalnylCoA_c → 4.0 CO2_c + 4.0 CoA_c + 2.0 H2O_c + 3.0 NADP_c + t4ketideCxl_c	*cxlA*
Cxl	rx0237	4.0 H_c + 2.0 MalnylCoA_c + 2.0 NADPH_c + t4ketideCxl_c → 2.0 CO2_c + 2.0 CoA_c + H2O_c + 2.0 NADP_c + h6ketideCxl_c	*cxlB*

Continuation of Table 2.6

Pathway[2]	Target	Reaction	Gene
Cxl	rx0238	4.0 H_c + MalnylCoA_c + 2.0 NADPH_c + SMylmalnylCoA_c + h6ketideCxl_c → 2.0 CO2_c + 2.0 CoA_c + H2O_c + 2.0 NADP_c + o8ketideCxl_c	*cxlC*
Cxl	rx0239	5.0 H_c + MalnylCoA_c + 3.0 NADPH_c + SMylmalnylCoA_c + o8ketideCxl_c → 2.0 CO2_c + 2.0 CoA_c + 2.0 H2O_c + 3.0 NADP_c + d10ketideCxl_c	*cxlD*
Cxl	rx0240	4.0 H_c + MalnylCoA_c + 2.0 NADPH_c + SMylmalnylCoA_c + d10ketideCxl_c → 2.0 CO2_c + 2.0 CoA_c + CxlA_c + 2.0 H2O_c + 2.0 NADP_c	*cxlE*
Cxl	rx1525	CxlA_c → CxlA_e	
Hybrid_PK-NP	rx0913	ATP_c + LThr_c → AMP_c + peptideHal_int_c + ppi_c	*sle09480*
Hybrid_PK-NP	rx0914	ATP_c + 4.0 H_c + LCys_c + 7.0 MalnylCoA_c + 4.0 NADPH_c + 2.0 O2_c + peptideHal_int_c → AMP_c + 8.0 CO2_c + 7.0 CoA_c + 5.0 H2O_c + 4.0 NADP_c + dipephexketide_c + ppi_c	*sle09500* and *sle09450* and *sle09440*
Hybrid_PK-NP	rx0915	dipephexketide_c → prohal_c	*sle09440* and *sle09420* and *sle09410* and *sle09510*
Hybrid_PK-NP	rx0916	prohal_c → hal_c	
Hybrid_PK-NP	rx1526	hal_c → hal_e	

[1]The gene targets are shown sorted by the compound predicted to be enhanced (colours).
[2]Chaxamycins (cxm); chaxalactins (cxl).

2.5 Discussion

2.5.1 Tools used for the reconstruction

In this work, *i*VR1007 was constructed de novo using bioinformatics tools. The construction of a GSM, without having a GSM as base, is labour intensive but it diminished possible errors due to outdated information and possible propagation of errors. The reconstruction of the metabolism of *S. leeuwenhoekii* C34 was facilitated by the development of the GeMRA interface that mainly uses python bioservices (Cokelaer et al., 2013) and COBRApy. There are bioinformatic tools that allow generating draft of GSMs like Model SEED (Devoid et al., 2013), however that platform is based in data available in SEED and since *i*VR1007 was constructed based on information of KEGG the information could differ.

Considering that *S. leeuwenhoekii* C34 was not included in KEGG database, since it was a novel strain, the comprehensive information of other *Streptomyces* was used to search for homologous genes and in that way identify enzymes of *S. leeuwenhoekii* C34 that participate in each reaction. It is critical to properly identify the homologous genes, iso-enzymes and enzymes complexes, since a wrong GPR relationship could lead to an incorrect interpretation of the optimization results.

The protein domain comparison tool developed allows to compare the domains of homologous proteins and determine if they were similar or not, preventing errors in the GPR relationship assignment.

By using the script developed in this work it is possible to easily compare the results of batch CDD search, and visualize which homologous proteins do not have the same domains. This procedure allowed identification of 14 genes that were wrongly assigned just based on homology.

The program developed allowed to create the GSM of *S. leeuwenhoekii* C34 from the annotated genome sequence. In addition, the python scripts developed in this work can be easily adapted to facilitate construction of other GSM, being a reliable alternative for start the construction of GSMs.

2.5.2 Discrepancies between model predictions and the experimental data

The model *i*VR1007 has a high accuracy, 83.7 %. This percentage of accuracy could be improved by analysing and understanding the discrepancies between predictions and experimental data. The model *i*VR1007 did not include regulatory constrains, hence the discrepancies could be due to missing regulatory information. The model could be improved with the inclusion of regulatory constrains (Herrgård et al., 2006). On the other hand, the identification of genes involved in reactions that were added as gaps would allowed to polish the GSM. For example, the GSM of *E. coli* had been highly improved through the years by the inclusion of experimental information (Orth et al., 2011).

GSM are always under development, and with the future availability of more experimental data for *S. leeuwenhoekii* C34, it is going to be possible to further improve the predictability of the model.

2.5.3 Essential genes of *S. leeuwenhoekii* C34

The predictability of essential genes has been addressed for *E. coli* using the great amount of information available for this microorganism, at it has been found to be of approximately 91 % (Joyce et al., 2006). Also, the accuracy of knockout prediction compared to gene essentiality data for *B. subtilis* was of 94 % (Oh et al., 2007) proving that GSM are a reliable tool for this type of analysis. So far there is no experimental information of essential genes for *S. leeuwenhoekii* C34, so the prediction of the model could not be compared. However, the predictions will be very useful when studying possible gene knockouts.

2.5.4 Identification of metabolic engineering targets for increasing chaxamycins, chaxalactins and hybrid PK-NP production

2.5.4.1 Prediction of gene knockout targets

The GSM allowed obtaining novel knockout gene targets that could be used to improve specialised metabolite biosynthesis (Table 2.3). Gene knockouts were detected that predict an increment in the production of chaxamycin A, chaxalactin A and hybrid PK-NP.

The deletion of *sle03600* produced an increased flux through the reactions rx0404 and rx0405 due to more availability of acetyl-CoA, and consequently is predicted an enhanced flux toward chaxalactin A production. Further deletion of *sle21250*, *sle22000* and *sle47020* (reaction rx0022), allows to increment the pool of acetyl-CoA that can go to the production of malonyl-CoA, a key

precursor in specialised metabolite biosynthesis.

The deletion of *sle39090* that participate in the reaction rx1350 that consumes UDP-glucose, in a Δ*sle03600* background, predict an increase in the fluxes toward the production of AHBA a key precursor for the biosynthesis of chaxamycin A. The production of hybrid PK-NP was predicted to be enhanced in a double mutant of Δ*sle03600* and Δ*sle47670*. The later deletion diminished the production of methylmalonyl-CoA that could be used as precursor for other specialised metabolites.

Similarly, other gene deletions predicted to increase chaxamycin A or hybrid PK-NP production were found to, directly or indirectly, increment in the fluxes toward the production of the building blocks of these metabolites. Such as an increment in the flux through the reactions rx0404 and rx0405, generating a higher conversion of acetyl-CoA to malonyl-CoA.

The knockout gene targets differ from previously reported ones and can lead to an improvement of specialised metabolite biosynthesis. Some of the reported gene knockouts for enhancing specialised metabolites in other *Streptomyces* are: *SCO1937* and *SCO6661* in *S. coelicolor* (Ryu et al., 2006). The homologous genes in *S. leeuwenhoekii* C34 are *sle52050* and *sle11620* that participate in the reactions rx1099 and rx1101. These genes were not found as deletion targets with *i*VR1007, and when the deletion of those genes was simulated, an increased production of specialised metabolites was not found, neither an increased flux through rx0404 and rx0405. However, using Δ*sle52050* and Δ*sle11620* as genetic background, it was possible to obtain other gene deletions that enhanced specialised metabolites production (data not shown).

It has been described that gene knockouts associated to the metabolism of N-acetyl-glucosamine produce a higher yield of actinorhodin (Swiatek et al., 2012). The homologous genes, *sle39040*, *sle25040*, *sle39050* and *sle43640*, are included in *i*VR1007, but they were not found as potential knockout targets. Nevertheless, these and other gene knockouts that were not found could be used in complementation to the predicted gene targets in order to boost specialised metabolites production.

Since gene targets were found for improving each specialised metabolite production separately, it is possible that the deletion of the predicted genes generate an increase in the production of any of the studied metabolites. Particularly, as many of the gene deletions cause an improvement in the flux through the reactions that convert acetyl-CoA to malonyl-CoA, the main precursor used to synthesised specialised metabolites. For this reason, it is important to consider the predicted gene deletions on modified strains that does not contain competitive BGC. For example, when improving production of chaxalactins or hybrid PK-NP the strain *S. leeuwenhoekii* M1614 (Δchaxamycins BGC) should be used as starter point.

2.5.4.2 Prediction of overexpression gene targets

Several potential gene targets for overexpression were identified using FSEOF. Nine reactions were identified as targets to increase the production of chaxamycins, chaxalactins or hybrid PK-NP, they are mainly associated with the biosynthesis of the precursor malonyl-CoA or with the pentose phosphate pathway (PPP).

An important precursor for the biosynthesis of polyketides such as chaxamycin A, chaxalactin A and hybrid PK-NP is malonyl-CoA. In the model, there are two essential reactions needed for malonyl-CoA biosynthesis, rx0404 and rx0405 (in KEGG: r04386 and r04385, respectively). These two reactions are usually lumped into one reaction (r00742) that was not included in the model to avoid duplicity. The genes predicted to catalyse both reactions are *sle47660* and any of the following: *sle27560*, *sle44630*, *sle39830* or *sle59710*. The two first could be better targets because they have higher homology to proteins already studied (Maharjan et al., 2010). However the protein Sle59710 despite of having lower identity to the associated proteins of the reaction rx0404, has the same domains that homologue proteins and also has the domains that are present in Sle47660. Further studies are required in order to determine if this protein is capable of catalyzing the conversion of acetyl-CoA to malonyl-CoA.

The reactions rx0404 and rx0405 were found as overexpression targets for all the cases. Overexpression of acetyl-CoA carboxylase (acc) has proven to be very useful to accomplish overproduction of several specialised metabolites (Ryu et al., 2006; Zha et al., 2009; Maharjan et al., 2010, 2012).

Besides the use of ATP and acetyl-CoA, for the generation of malonyl-CoA in reaction rx0404 and rx0405, it is also needed bicarbonate. Bicarbonate is generated from CO_2 in the reaction rx1057 (KEGG No. r10092). This reaction was found as a target for the three specialised metabolites, and the three genes associated are *sle16220*, *sle32790* and *sle50480*. The overexpression of homologous genes of other strains had not been required when overexpressing acetyl-CoA carboxylase complex (Ryu et al., 2006; Zha et al., 2009; Maharjan et al., 2010, 2012), this indicates that the bicarbonate is not restricting the reaction, therefore the overexpression of the genes in *S. leeuwenhoekii* C34 could not affect the specialised metabolite production.

Several targets are associated to the PPP and some of them were identified as targets for the three specialised metabolites (rx1090 and rx1095), and other for chaxalactins and hybrid PK-NP (rx1094, rx1098 and rx1103) and one just for chaxalactins (rx1226). The reaction rx1090 (KEGG No. r01827) associated to the genes *sle11610* or *sle52060*, produces D-erythrose 4-phosphate and β-D-fructose 6-phosphate. The interlink between the PPP and the glycolysis pathway produces that any modifications in either of the pathways directly affect the other one (Olano et al., 2008). In the case of the modifications suggested by the model, the higher flux of rx1090 could generated a higher flux of β-D-fructose 6-phosphate to the glycolysis pathway.

There are other reactions (rx0291, rx1711, rx0288, rx1709, rx1710, rx0283 and rx0264) that are associated to the cysteine and methionine metabolism. They seem to be selected as targets for overexpression since they could increase the acetyl-CoA pool (rx0288, in KEGG r00586), however this reaction is a gap in the model. There is not evidence of overexpression of homologous genes in other strains to increase specialised production. Hence, the overexpression of any of the associated genes should be addressed carefully.

Several of the reactions that have an increased flux while enforcing hybrid PK-NP production are also increased when enforcing production of chaxamycins. So, it is clear that the biosynthesis pathway of this two specialised metabolites are competing for precursors. For the same reason any

overexpression mutant to accomplish an increased production of hybrid PK-NP should be done in the Δ chaxamycins BGC strain (M1614). Also, it would be better if the proposed modifications could be performed in a *S. leeuwenhoekii* C34 strain lacking the main active BGCs. Therefore, the metabolic profile would be easy to analyse and there would be more precursors available for the biosynthesis of the desired specialised metabolite (Gomez-Escribano and Bibb, 2011).

Amongst the reactions not directly involved that were found as targets to enhance production of chaxamycins, there is the reaction rx1367 (KEGG No. r00289) that uses D-glucose 1-phosphate to produce UDP-glucose, the later compound is needed for the biosynthesis of AHBA. The gene associated to this reaction is *sle41020*. The expression of the homologous gene, *galU*, associated to rifamycin biosynthesis had been increased by the incorporation of nitrate to the media (Shao et al., 2015). However, it has not been study the effect of the overexpression under a constitutive strong promoter over the AHBA production.

The genes associated to the reaction rx0227 (KEGG No. r00833), *sle28060* or *sle28760* or *sle22410* or *sle22940*, are also targets for overexpression since the production of (R)-methylmalonyl-CoA is predicted to be used to produced (S)-methylmalonyl-CoA by the gen *sle22570* in the reaction rx0228 (KEGG No. r02765). (S)-methylmalonyl-CoA is an important precursor in the biosynthesis of chaxamycins and chaxalactins, therefore these overexpression targets were found for both cases. Overexpression of homologous genes in other *Streptomyces* had not been performed. The overexpression of this genes should be studied in conjunction with the overexpression of the genes predicted to increase the biosynthesis malonyl-CoA and AHBA.

Amongst the other common overexpression targets identified for chaxamycins and chaxalactins, there is a set of reactions (rx1257, rx1258, rx1262, rx1263, rx1277 and rx1628) that lead to the production of glyoxylate and urea, however the last reaction is a gap in the model. Therefore is necessary to accomplish more studies related to the identification of enzymes that could be associated to that reaction.

From the gene targets identified for overexpression other genes that could be used could be inferred. For example, it can be inferred that succinyl-CoA is required for the production of (S)-methylmalonyl-CoA, a precursor molecule of chaxamycins and chaxalactins. However, the FSEOF result does not include overexpression targets for increasing succinyl-CoA because there is more than one reaction from where the model can predict succinyl-CoA production, like rx0205 or rx1000 (in KEGG: r01197 and r08549, respectively). In the case of overexpressing those reactions it will be required to overexpress the genes: *sle30310* with *sle30300* (for rx0205) and *sle24580* with *sle49590* and with *sle49600* (for rx1000). A similar case can be proposed for the case of the reaction rx1551 (KEGG No. r00678) that generate L-formylkynurenine that could be used to produce L-alanine in the reaction rx1557 (KEGG No. r03936) this could be used later to produce pyruvate in the reaction rx0049 (KEGG No. r00396). In this case, the two last reactions were not predicted as overexpression targets because there are several reactions in which the involved metabolites participate. The gene encoding for the enzyme catalysing the reaction rx1551, *sle33790*, could be used as a non-intuitive overexpression target.

Several of the reactions identified are associated with isoenzymes, so it is crucial to analyse

through RT-PCR which genes are actually being expressed (see section 2.5.5).

All the predicted overexpression targets should be used for metabolic engineering of *S. leeuwenhoekii* C34. Also, the overexpression targets can be included as gene targets for improving PKS production identified in similar strains. On the other hand, the gene overexpression targets identified for other strains elsewhere, that were not detected in this study, could also be considered as targets. Such is the case of the overexpression of the branched-chain α-keto acid dehydrogenase (BCDH) complex that improves actinorhodin production in *S. coelicolor* (Kim et al., 2014).

2.5.5 Addressing functionality and genetic redundancy

The genomes of *Streptomyces* strains usually contains several genes encoding for the same biochemical function (Bentley et al., 2002; Ikeda et al., 2003). This has also been observed for *S. leeuwenhoekii* C34 in this work. The study of the essential genes of the central metabolism in *E. coli* (Kim and Copley, 2007) showed that a large number (80 out of 227) were nonessential under the studied conditions. The reason for this is the availability of alternative pathways, isoenzymes, multifunctional enzymes or broad-specific enzymes (Kim and Copley, 2007). The redundancy or metabolic flexibility, allows the cell to use different pathways under specific environmental conditions (Hiltner et al., 2015).

In the GSMs the isoenzymes are associated with a particular pathway. However, it is important to study their regulation to determine their biochemical relevance under different culture conditions. Also, to evaluate their impact in the production of specialised metabolites.

2.5.6 The genome scale model of *S. leeuwenhoekii* C34 would allow experimental design

The combination of the genes identified as targets of overexpression and knockouts can be used to design strains with specific phenotypes. Also, the model can be applied to identify the formulation of an optimal media for the production of each specialised metabolite. Similarly, the model should be used to study gene deletions that would help in understanding and complementing the biosynthesis pathways that, currently, are incomplete.

2.6 Future work

The next step will be to perform metabolic engineering using the gene deletions or/and overexpression suggested by *i*VR1007 that should enhance production of specialised metabolites. In the case that more than one gene is associated to a reaction, RT-PCR will be used to identify the genes that are being expressed. If more than one gene is being expressed, all of them will be deleted/overexpressed. Deletion of a gene through standard techniques is a very tedious work that could take from one to several months. The new technique of CRISPR-Cas9 has been proven to be useful for quick generation of mutants of different species, specifically it has been used to accomplish gene deletion

in *Streptomyces* with high accuracy (Cobb et al., 2014). The CRISPR-Cas9 system optimized to work in *Streptomyces* strains is currently being tested for *S. leeuwenhoekii* C34 by co-workers, and if it is successful it will be used to generate the deletion/overexpression of genes suggested in this work.

The model *i*VR1007 will be improved by using information of other carbon, nitrogen, phosphorous and sulphur sources, enzymes activities, inclusion of other specialised metabolites and a more comprehensive study of the incomplete pathways.

2.7 Conclusions

1. An interface that helps in the reconstruction of GSMs was developed.

2. The first GSM of *S. leeuwenhoekii* C34, *i*VR1007, has been constructed and it had 83.7 % of accuracy.

3. The deletion of the genes *sle03600* and *sle21250*, *sle22000* and *sle47020* predicted an increase in chaxalactins production.

4. The chaxamycin production was predicted to be enhanced by the knockout of *sle03600* and *sle39090* due to the indirect increment of malonyl-CoA and AHBA production, respectively.

5. The hybrid PK-NP production was predicted to be incremented by the deletion of *sle03600* and *sle47670*.

6. Thirty-five, 25 and 27 gene overexpression targets were predicted for enhancing the production of chaxamycins, chaxalactins and hybrid PK-NP, respectively, that were not directly related to their biosynthesis.

Wrapping up: general conclusions and perspectives

The main objective of this thesis was to study the metabolism of *S. leeuwenhoekii* C34 with focus in specialised metabolite biosynthesis pathways and to find metabolic engineering targets that would increase their production. To accomplish this objective, four specific objectives were defined:

1. Identification of biosynthesis gene clusters of specialised metabolites in *S. leeuwenhoekii* C34 by genome mining.

2. To study the product of a novel biosynthetic gene cluster *in vivo*.

3. To develop a genome scale model of *S. leeuwenhoekii* C34 to analyse its metabolism.

4. To identify metabolic engineering targets that would enhance the production of a specialised metabolite, like chaxamycins, in *S. leeuwenhoekii* C34.

In the first chapter of this thesis, the identification of 34 BGCs for specialised metabolites encoded in the chromosome of *S. leeuwenhoekii*, plus the identification of one BGC encoded in the plasmid pSLE2 was presented. The bioinformatic evaluation of these BGCs revealed that twenty-seven were probably involved in the synthesis of new specialised metabolites. The number of BGCs is similar to that found in other *Streptomyces* strains. Several of the BGCs are encoding for polyketides, non-ribosomal peptides or hybrid systems. Specifically, it was studied *in vivo* the product of one lasso-peptide BGC and a hybrid *trans*-AT PKS/NRPS BGC.

The lasso-peptide 3 was found in the supernatant of 5 day liquid cultures. The BGC of lasso-peptide 3 was successfully cloned and expressed in the heterologous host *S. coelicolor* M1152. Further analysis of this lasso-peptide allowed to confirmation that it was equal to the chaxapeptin, a recently described lasso-peptide synthesised by *S. leeuwenhoekii* C58.

The study of the hybrid *trans*-AT PKS/NRPS BGC required further efforts. This BGC was bioinformatically compared to the leinamycin BGC, similar characteristics were found in both BGCs, thus the available information of leinamycin biosynthesis together with the fingerprint information

of the PKS/NRPS domains of the hybrid *trans*-AT PKS/NRPS biosynthetic genes was used to predict the base structure of the hybrid *trans*-AT PKS/NRPS BGC product. The predicted structure did not include the action of post-synthesis modification genes, such as *sle09470* that encodes for a chlorinating enzyme.

Moreover, several strains with modification to the hybrid *trans*-AT PKS/NRPS BGC were developed. For example, in the strain *S. leeuwenhoekii* M1601, the key biosynthetic gene *sle09500* encoding for the *trans*-AT domain was deleted. Strains that lacks post-modification genes were also developed, such as the gene *sle09470*. The deletion or overexpression of regulatory genes was addressed. The deletion of chaxamycins BGC was done in order to increase the pool of precursors available for the biosynthesis of specialised metabolites obtaining the strain *S. leeuwenhoekii* M1614. In addition to the deletion of the chaxamycin BGC it was deleted the regulatory gene *sle09560* (strain M1619). On the other hand, the hybrid *trans*-AT PKS/NRPS BGC was cloned in the heterologous hosts *S. coelicolor* M1152, *S. coelicolor* M1154 and *S. albus* J1074.

Liquid cultures in seed and production media were performed for the different strains developed in this work. The supernatant and mycelium extract samples during the cultures were analysed by HPLC MS/MS. Also, the antibiotic activity of the samples was evaluated in bioassays against the bacteria *B. subtilis*.

Metabolite profile comparison of samples analysed by HPLC MS/MS was performed. Compounds with halogenated pattern were identified, however they were unlikely to be product of the studied BGC since they were not unique to *S. leeuwenhoekii* strains. An ion m/z 611.53 $[M + H]^+$ that does not had a chlorinated pattern was identified as a possible product of the hybrid *trans*-AT PKS/NRPS BGC since it was only detected in samples of *S. leeuwenhoekii* M1614 and *S. leeuwenhoekii* M1619, and with more intensity in the latter. The deletion of the biosynthetic gene *sle09500* in the strain *S. leeuwenhoekii* M1614 and M1619 would show if the detected ion is a product of the hybrid *trans*-AT PKS/NRPS BGC. On the other hand, for detection of a chlorinated product it would be necessary to use ICP-MS and/or to overexpress the halogenating enzyme (*sle09470*) in the mentioned strains.

In addition, the overexpression of *sle09570* (efflux pump) could be performed in order to evaluate if is associated to the transport of the hybrid PK-NP and therefore if it would improve its production.

In the samples from taken from the heterologous hosts cultures containing the hybrid *trans*-AT PKS/NRPS BGC it was not possible to detect a differential peak through metabolite profile comparison. However, it could be considered to perform the same deletions/overexpression studied in the native strain in order to accomplish the heterologous expression.

In the second chapter of this thesis, the development of the first GSM of *S. leeuwenhoekii* C34 was described. The GSM, designated *i*VR1007, was constructed with the aid of a python based interface. The interface, GeMRA, allows retrieval of information from KEGG database, construct the GSM and performance of flux balance simulations, among other things. This tool could be

adapted to the construction of other GSMs and is user-friendly.

Experimental information of growth in different carbon, nitrogen and phosphorous sources was used to validate the model *i*VR1007. The model showed 83.7 % of accuracy. The discrepancies between the model predictions and experimental data was attributed to errors, inconsistencies or missing constrains, such as transcriptional regulatory constraints. For allowing the model to use some carbon/nitrogen sources it was necessary to include gap reactions, further studies are required to understand the pathways that allowed the uses of those carbon/nitrogen sources experimentally.

Metabolic engineering targets for increasing the production of specialised metabolites were predicted using the model *i*VR1007. The production of chaxalactins was predicted to be incremented by the deletion of the genes *sle03600* and *sle21250*, *sle22000* and *sle47020*. The deletion of those genes prevent the consumption of acetyl-CoA in reactions that are not associated to the specialised metabolites biosynthesis, therefore increasing the acetyl-CoA pool. The more availability of acetyl-CoA allows production of more malonyl-CoA, an important building block of specialised metabolites biosynthesis.

The chaxamycins production was predicted to be enhanced by the deletion of *sle03600* and *sle39090* due to the indirect increment of malonyl-CoA and AHBA production, respectively. Further improvement was predicted with the additional deletion of the genes *sle29000*, *sle29010*, and *sle56410* associated to the reaction rx1231.

The hybrid PK-NP production was predicted to be increased by the deletion of *sle03600* and *sle47670*, the later deletion diminished the production of (S)-methylmalonyl-CoA promoting the production of specialised metabolites that not require that metabolite as building block, as the hybrid PK-NP. This gene deletion could be performed in addition to the previous modifications done to *S. leeuwenhoekii* C34 WT. Also, the targets predicted for overexpression by the model should be considered.

Gene targets for overexpression were found for each of the analysed specialised metabolites. Specifically, 35, 25 and 27 overexpression targets were predicted for enhancing the production of chaxamycins, chaxalactins, and hybrid PK-NP, respectively. These targets were not directly related to their biosynthesis. The majority of them correspond to new overexpression targets, although homologous genes to previously described overexpression targets for increasing other polyketides biosynthesis were also found, as the genes associated with the conversion of acetyl-CoA to malonyl-CoA in the reactions rx0404 and rx0405.

The metabolic engineering modifications suggested by the GSM predictions for enhancing the specialised metabolite production should be performed in a *S. leeuwenhoekii* C34 strain lacking the BGC of the not wanted metabolites. Therefore, the modifications would redirect the fluxes to the production of the desired specialised metabolites and there will be more availability of precursors for its biosynthesis.

Bibliography

Alam, M. T., Merlo, M. E., Takano, E., and Breitling, R. (2010). Genome-based phylogenetic analysis of *Streptomyces* and its relatives. *Molecular Phylogenetics and Evolution*, 54:763–772.

Alt, S. and Wilkinson, B. (2015). Biosynthesis of the Novel Macrolide Antibiotic Anthracimycin. *ACS Chemical Biology*, 10:2468–2479.

Alting-Mees, M. and Short, J. (1989). pBluescript II: gene mapping vectors. *Nucleic Acids Research*, 17:9494.

Aziz, R. K., Bartels, D., Best, A. A., DeJongh, M., Disz, T., Edwards, R. A., Formsma, K., Gerdes, S., Glass, E. M., Kubal, M., et al. (2008). The RAST Server: rapid annotations using subsystems technology. *BMC Genomics*, 9:75.

Baranašić, D., Zucko, J., Diminic, J., Gacesa, R., Long, P. F., Cullum, J., Hranueli, D., and Starcevic, A. (2014). Predicting substrate specificity of adenylation domains of nonribosomal peptide synthetases and other protein properties by latent semantic indexing. *Journal of Industrial Microbiology & Biotechnology*, 41:461–467.

Barona-Gómez, F., Wong, U., Giannakopulos, A. E., Derrick, P. J., and Challis, G. L. (2004). Identification of a cluster of genes that directs desferrioxamine biosynthesis in *Streptomyces coelicolor* M145. *Journal of the American Chemical Society*, 126:16282–16283.

Baumeister, R., Helbl, V., and Hillen, W. (1992). Contacts between Tet repressor and tet operator revealed by new recognition specificities of single amino acid replacement mutants. *Journal of molecular biology*, 226:1257–1270.

Bentley, S., Chater, K., Cerdeno-Tarraga, A.-M., Challis, G., Thomson, N., James, K., Harris, D., Quail, M., Kieser, H., Harper, D., et al. (2002). Complete genome sequence of the model actinomycete *Streptomyces coelicolor* A3 (2). *Nature*, 417:141–147.

Bergmann, S., Schümann, J., Scherlach, K., Lange, C., Brakhage, A. A., and Hertweck, C. (2007). Genomics-driven discovery of PKS-NRPS hybrid metabolites from *Aspergillus nidulans*. *Nature Chemical Biology*, 3:213–217.

Bibb, M. J. (2005). Regulation of secondary metabolism in streptomycetes. *Current Opinion in Microbiology*, 8:208–215.

Blin, K., Medema, M. H., Kazempour, D., Fischbach, M. A., Breitling, R., Takano, E., and Weber, T. (2013). antiSMASH 2.0—a versatile platform for genome mining of secondary metabolite producers. *Nucleic Acids Research*, page gkt449.

Blum, R. H., Carter, S. K., and Agre, K. (1973). A clinical review of bleomycin—a new antineoplastic agent. *Cancer*, 31:903–914.

Bode, H. B., Bethe, B., Höfs, R., and Zeeck, A. (2002). Big effects from small changes: possible ways to explore nature's chemical diversity. *ChemBioChem*, 3:619–627.

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

Bu, X., Wang, T., and Hall, G. (2003). Determination of halogens in organic compounds by high resolution inductively coupled plasma mass spectrometry (HR-ICP-MS). *Journal of Analytical Atomic Spectrometry*, 18:1443–1451.

Bursy, J., Kuhlmann, A. U., Pittelkow, M., Hartmann, H., Jebbar, M., Pierik, A. J., and Bremer, E. (2008). Synthesis and uptake of the compatible solutes ectoine and 5-hydroxyectoine by *Streptomyces coelicolor* A3 (2) in response to salt and heat stresses. *Applied and Environmental Microbiology*, 74:7286–7296.

Busarakam, K. (2014). *Novel Actinobacterial Diversity in Arid Atacama Desert Soils as a Source of New Drug Leads*. PhD thesis, Newcastle University.

Busarakam, K., Bull, A. T., Girard, G., Labeda, D. P., van Wezel, G. P., and Goodfellow, M. (2014). *Streptomyces leeuwenhoekii* sp. nov., the producer of chaxalactins and chaxamycins, forms a distinct branch in *Streptomyces* gene trees. *Antonie van Leeuwenhoek*, 105:849–861.

Caffrey, P. (2003). Conserved amino acid residues correlating with ketoreductase stereospecificity in modular polyketide synthases. *ChemBioChem*, 4:654–657.

Calderone, C. T. (2008). Isoprenoid-like alkylations in polyketide biosynthesis. *Natural Product Reports*, 25:845–853.

Campodonico, M. A., Andrews, B. A., Asenjo, J. A., Palsson, B. O., and Feist, A. M. (2014). Generation of an atlas for commodity chemical production in *Escherichia coli* and a novel pathway prediction algorithm, GEM-Path. *Metabolic Engineering*, 25:140–158.

Campodonico, M. A., Vaisman, D., Castro, J. F., Razmilic, V., Mercado, F., Andrews, B. A., Feist, A. M., and Asenjo, J. A. (2016). *Acidithiobacillus ferrooxidans*'s comprehensive model driven analysis of the electron transfer metabolism and synthetic strain design for biomining applications. *Metabolic Engineering Communications*, 3:84–96.

Castro, J. F. (2015). *Identification of the chaxamycin and chaxalactin biosynthesis genes through genome mining of Streptomyces leeuwenhoekii C34 and heterologous production of chaxamycins in Streptomyces coelicolor M1152*. PhD thesis, University Of Chile.

Castro, J. F., Razmilic, V., Gomez-Escribano, J. P., Andrews, B., Asenjo, J. A., and Bibb, M. J. (2015). Identification and heterologous expression of the chaxamycin biosynthesis gene cluster from *Streptomyces leeuwenhoekii*. *Applied and environmental microbiology*, 81:5820–5831.

Challis, G. L. (2008a). Genome mining for novel natural product discovery. *Journal of Medicinal Chemistry*, 51:2618–2628.

Challis, G. L. (2008b). Mining microbial genomes for new natural products and biosynthetic pathways. *Microbiology*, 154:1555–1569.

Challis, G. L. and Naismith, J. H. (2004). Structural aspects of non-ribosomal peptide biosynthesis. *Current Opinion in Structural Biology*, 14:748–756.

Cheng, Y.-Q., Tang, G.-L., and Shen, B. (2002). Identification and localization of the gene cluster encoding biosynthesis of the antitumor macrolactam leinamycin in *Streptomyces atroolivaceus* S-140. *Journal of Bacteriology*, 184:7013–7024.

Cheng, Y.-Q., Tang, G.-L., and Shen, B. (2003). Type I polyketide synthase requiring a discrete acyltransferase for polyketide biosynthesis. *Proceedings of the National Academy of Sciences*, 100:3149–3154.

Choi, H. S., Lee, S. Y., Kim, T. Y., and Woo, H. M. (2010). *In silico* identification of gene amplification targets for improvement of lycopene production. *Applied and Environmental Microbiology*, 76:3097–3105.

Chooi, Y.-H., Fang, J., Liu, H., Filler, S. G., Wang, P., and Tang, Y. (2013). Genome mining of a prenylated and immunosuppressive polyketide from pathogenic fungi. *Organic letters*, 15(4):780–783.

Cobb, R. E., Wang, Y., and Zhao, H. (2014). High-efficiency multiplex genome editing of *Streptomyces* sp.cies using an engineered CRISPR/Cas system. *ACS Synthetic Biology*, 4:723–728.

Cokelaer, T., Pultz, D., Harder, L. M., Serra-Musach, J., and Saez-Rodriguez, J. (2013). BioServices: a common Python package to access biological Web Services programmatically. *Bioinformatics*, 29:3241–3242.

Contador, C., Rodríguez, V., Andrews, B., and Asenjo, J. (2015). Genome-scale reconstruction of *Salinispora tropica* CNB-440 metabolism to study strain-specific adaptation. *Antonie van Leeuwenhoek*, 108:1075–1090.

Cragg, G. and Newman, D. (2009). Nature: a vital source of leads for anticancer drug development. *Phytochemistry Reviews*, 8:313–331.

Davies, J. and Davies, D. (2010). Origins and evolution of antibiotic resistance. *Microbiology and Molecular Biology Reviews*, 74:417–433.

Demain, A. (1999). Pharmaceutically active secondary metabolites of microorganisms. *Applied Microbiology and Biotechnology*, 52:455–463.

Demain, A. L. and Fang, A. (2000). The natural functions of secondary metabolites. In *History of Modern Biotechnology I*, pages 1–39. Springer.

Devoid, S., Overbeek, R., DeJongh, M., Vonstein, V., Best, A. A., and Henry, C. (2013). Automated genome annotation and metabolic model reconstruction in the SEED and Model SEED. *Systems Metabolic Engineering: Methods and Protocols*, pages 17–45.

D'Huys, P.-J., Lule, I., Vercammen, D., Anné, J., Van Impe, J. F., and Bernaerts, K. (2012). Genome-scale metabolic flux analysis of *Streptomyces lividans* growing on a complex medium. *Journal of Biotechnology*, 161:1–13.

Djinni, I., Defant, A., Kecha, M., and Mancini, I. (2014). Metabolite profile of marine-derived endophytic *Streptomyces sundarbansensis* WR1L1S8 by liquid chromatography–mass spectrometry and evaluation of culture conditions on antibacterial activity and mycelial growth. *Journal of applied microbiology*, 116:39–50.

Du, L., Sánchez, C., and Shen, B. (2001). Hybrid peptide–polyketide natural products: biosynthesis and prospects toward engineering novel molecules. *Metabolic Engineering*, 3:78–95.

Ebrahim, A., Lerman, J. A., Palsson, B. O., and Hyduke, D. R. (2013). COBRApy: constraints-based reconstruction and analysis for python. *BMC Systems Biology*, 7:74.

El-Sayed, A. K., Hothersall, J., Cooper, S. M., Stephens, E., Simpson, T. J., and Thomas, C. M. (2003). Characterization of the mupirocin biosynthesis gene cluster from *Pseudomonas fluorescens* NCIMB 10586. *Chemistry & Biology*, 10:419–430.

Elsayed, S. S., Trusch, F., Deng, H., Raab, A., Prokes, I., Busarakam, K., Asenjo, J. A., Andrews, B. A., Van West, P., Bull, A. T., et al. (2015). Chaxapeptin, a lasso peptide from extremotolerant *Streptomyces leeuwenhoekii* strain C58 from the hyperarid Atacama desert. *The Journal of Organic Chemistry*, 80:10252–10260.

Finking, R. and Marahiel, M. A. (2004). Biosynthesis of nonribosomal peptides 1. *Annual Review of Microbiology*, 58:453–488.

Flamholz, A., Noor, E., Bar-Even, A., and Milo, R. (2011). eQuilibrator—the biochemical thermodynamics calculator. *Nucleic Acids Research*, page gkr874.

Fleming, A. (1929). On the antibacterial action of cultures of a penicillium, with special reference to their use in the isolation of *B. influenzae*. *British Journal of Experimental Pathology*, 10:226.

Frisvad, J. C. (1987). High-performance liquid chromatographic determination of profiles of mycotoxines and other secondary metabolites. *Journal of Chromatography A*, 392:333–347.

Gates, K. S. (2000). Mechanisms of DNA damage by leinamycin. *Chemical Research in Toxicology*, 13:953–956.

Gomez-Escribano, J. P. and Bibb, M. J. (2011). Engineering *Streptomyces coelicolor* for heterologous expression of secondary metabolite gene clusters. *Microbial Biotechnology*, 4:207–215.

Gomez-Escribano, J. P. and Bibb, M. J. (2014). Heterologous expression of natural product biosynthetic gene clusters in *Streptomyces coelicolor*: from genome mining to manipulation of biosynthetic pathways. *Journal of industrial microbiology & biotechnology*, 41:425–431.

Gomez-Escribano, J. P., Castro, J. F., Razmilic, V., Chandra, G., Andrews, B., Asenjo, J. A., and Bibb, M. J. (2015). The *Streptomyces leeuwenhoekii* genome: de novo sequencing and assembly in single contigs of the chromosome, circular plasmid pSLE1 and linear plasmid pSLE2. *BMC Genomics*, 16:485.

Gribble, G. W. (2004). Natural organohalogens: a new frontier for medicinal agents? *Journal of Chemical Education*, 81:1441.

Gross, H. (2007). Strategies to unravel the function of orphan biosynthesis pathways: recent examples and future prospects. *Applied Microbiology and Biotechnology*, 75:267–277.

Günter, K., Toupet, C., and Schupp, T. (1993). Characterization of an iron-regulated promoter involved in desferrioxamine B synthesis in *Streptomyces pilosus*: repressor-binding site and homology to the diphtheria toxin gene promoter. *Journal of Bacteriology*, 175:3295–3302.

Haines, A. S., Dong, X., Song, Z., Farmer, R., Williams, C., Hothersall, J., Płoskoń, E., Wattana-amorn, P., Stephens, E. R., Yamada, E., et al. (2013). A conserved motif flags acyl carrier proteins for β-branching in polyketide synthesis. *Nature Chemical Biology*, 9:685–692.

Hanahan, D. (1983). Studies on transformation of *Escherichia coli* with plasmids. *Journal of Molecular Biology*, 166:557–580.

Hara, M., Saitoh, Y., and Nakano, H. (1990). DNA strand scission by the novel antitumor antibiotic leinamycin. *Biochemistry*, 29:5676–5681.

Hara, M., Takahashi, I., Yoshida, M., Asano, K., Kawamoto, I., Morimoto, M., and Nakano, H. (1989). DC 107, a novel antitumor antibiotic produced by a *Streptomyces* sp. *The Journal of Antibiotics*, 42:333–335.

Harris, C. M., Kannan, R., Kopecka, H., and Harris, T. M. (1985). The role of the chlorine substituents in the antibiotic vancomycin: preparation and characterization of mono-and didechlorovancomycin. *Journal of the American Chemical Society*, 107:6652–6658.

Helfrich, E. J. and Piel, J. (2016). Biosynthesis of polyketides by *trans*-AT polyketide synthases. *Natural product reports*, 33:231–316.

Herrgård, M. J., Lee, B.-S., Portnoy, V., and Palsson, B. Ø. (2006). Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in *Saccharomyces cerevisiae*. *Genome research*, 16:627–635.

Hiltner, J. K., Hunter, I. S., and Hoskisson, P. A. (2015). Chapter Four-Tailoring specialized metabolite production in *Streptomyces*. *Advances in Applied Microbiology*, 91:237–255.

Hobbs, G., Frazer, C. M., Gardner, D. C., Cullum, J. A., and Oliver, S. G. (1989). Dispersed growth of *Streptomyces* in liquid culture. *Applied Microbiology and Biotechnology*, 31:272–277.

Hong, H.-J., Hutchings, M. I., Hill, L. M., and Buttner, M. J. (2005). The role of the novel Fem protein VanK in vancomycin resistance in *Streptomyces coelicolor*. *Journal of Biological Chemistry*, 280:13055–13061.

Hopwood, D. A. (2007). How do antibiotic-producing bacteria ensure their self-resistance before antibiotic biosynthesis incapacitates them? *Molecular microbiology*, 63:937–940.

Huang, D., Li, S., Xia, M., Wen, J., and Jia, X. (2013). Genome-scale metabolic network guided engineering of *Streptomyces tsukubaensis* for FK506 production improvement. *Microbial Cell Factories*, 12:1.

Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L., Larimer, F. W., and Hauser, L. J. (2010). Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 11:1.

Ikeda, H., Ishikawa, J., Hanamoto, A., Shinose, M., Kikuchi, H., Shiba, T., Sakaki, Y., Hattori, M., and Ōmura, S. (2003). Complete genome sequence and comparative analysis of the industrial microorganism *Streptomyces avermitilis*. *Nature Biotechnology*, 21:526–531.

Ikeda, H., Shin-ya, K., and Omura, S. (2014). Genome mining of the *Streptomyces avermitilis* genome and development of genome-minimized hosts for heterologous expression of biosynthetic gene clusters. *Journal of Industrial Microbiology & Biotechnology*, 41:233–250.

Ingraham, J. L., Maaløe, O., Neidhardt, F. C., et al. (1983). *Growth of the bacterial cell*. Sinauer, Sunderland, MA.

Inokoshi, J., Matsuhama, M., Miyake, M., Ikeda, H., and Tomoda, H. (2012). Molecular cloning of the gene cluster for lariatin biosynthesis of *Rhodococcus jostii* K01-B0171. *Applied Microbiology and Biotechnology*, 95:451–460.

Jankowitsch, F., Schwarz, J., Rückert, C., Gust, B., Szczepanowski, R., Blom, J., Pelzer, S., Kalinowski, J., and Mack, M. (2012). Genome sequence of the bacterium *Streptomyces davawensis* JCM 4913 and heterologous production of the unique antibiotic roseoflavin. *Journal of Bacteriology*, 194:6818–6827.

Joyce, A. R., Reed, J. L., White, A., Edwards, R., Osterman, A., Baba, T., Mori, H., Lesely, S. A., Palsson, B. Ø., and Agarwalla, S. (2006). Experimental and computational assessment of conditionally essential genes in *Escherichia coli*. *Journal of Bacteriology*, 188:8259–8271.

Kanehisa, M. and Goto, S. (2000). KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28:27–30.

Kanehisa, M., Goto, S., Sato, Y., Kawashima, M., Furumichi, M., and Tanabe, M. (2014). Data, information, knowledge and principle: back to metabolism in KEGG. *Nucleic Acids Research*, 42:D199–D205.

Keatinge-Clay, A. T. (2012). The structures of type I polyketide synthases. *Natural Product Reports*, 29:1050–1073.

Kersten, W. and Kersten, H. (1965). The binding of daunomycin, cinerubin and chromomycin A3 to nucleic acids. *Biochemische Zeitschrift*, 341:174.

Kieser, T., Bibb, M., Buttner, M., Chater, K., and Hopwood, D. (2000). *Practical Streptomyces manual*. The John Innes Foundation, Norwich, UK, second edition.

Kim, H. B., Smith, C. P., Micklefield, J., and Mavituna, F. (2004). Metabolic flux analysis for calcium dependent antibiotic (CDA) production in *Streptomyces coelicolor*. *Metabolic Engineering*, 6:313–325.

Kim, J. and Copley, S. D. (2007). Why metabolic enzymes are essential or nonessential for growth of *Escherichia coli* K12 on glucose. *Biochemistry*, 46:12501–12511.

Kim, M., Sang Yi, J., Kim, J., Kim, J.-N., Kim, M. W., and Kim, B.-G. (2014). Reconstruction of a high-quality metabolic model enables the identification of gene overexpression targets for enhanced antibiotic production in *Streptomyces coelicolor* A3 (2). *Biotechnology Journal*, 9:1185–1194.

Kim, T. Y., Sohn, S. B., Kim, Y. B., Kim, W. J., and Lee, S. Y. (2012). Recent advances in reconstruction and applications of genome-scale metabolic models. *Current Opinion in Biotechnology*, 23:617–623.

King, Z. A., Dräger, A., Ebrahim, A., Sonnenschein, N., Lewis, N. E., and Palsson, B. O. (2015). Escher: a web application for building, sharing, and embedding data-rich visualizations of biological pathways. *PLoS Comput Biol*, 11:e1004321.

Kobayashi, J. and Ishibashi, M. (1993). Bioactive metabolites of symbiotic marine microorganisms. *Chemical Reviews*, 93:1753–1769.

Laureti, L., Song, L., Huang, S., Corre, C., Leblond, P., Challis, G. L., and Aigle, B. (2011). Identification of a bioactive 51-membered macrolide complex by activation of a silent polyketide synthase in *Streptomyces ambofaciens*. *Proceedings of the National Academy of Sciences*, 108:6258–6263.

Lautru, S., Deeth, R. J., Bailey, L. M., and Challis, G. L. (2005). Discovery of a new peptide natural product by *Streptomyces coelicolor* genome mining. *Nature Chemical Biology*, 1:265–269.

Lin, X., Hopson, R., and Cane, D. E. (2006). Genome mining in *Streptomyces coelicolor*: molecular cloning and characterization of a new sesquiterpene synthase. *Journal of the American Chemical Society*, 128:6022–6023.

Liu, T., Huang, Y., and Shen, B. (2009). Bifunctional acyltransferase/decarboxylase LnmK as the missing link for β-alkylation in polyketide biosynthesis. *Journal of the American Chemical Society*, 131:6900–6901.

Lohman, J. R., Bingman, C. A., Phillips Jr, G. N., and Shen, B. (2013). Structure of the bifunctional acyltransferase/decarboxylase LnmK from the leinamycin biosynthetic pathway revealing novel activity for a double-hot-dog fold. *Biochemistry*, 52:902–911.

Ma, M., Lohman, J. R., Liu, T., and Shen, B. (2015). C-S bond cleavage by a polyketide synthase domain. *Proceedings of the National Academy of Sciences*, 112:10359–10364.

Maharjan, S., Koju, D., Lee, H. C., Yoo, J. C., and Sohng, J. K. (2012). Metabolic engineering of *Nocardia* sp. CS682 for enhanced production of nargenicin A1. *Applied Biochemistry and Biotechnology*, 166:805–817.

Maharjan, S., Park, J. W., Yoon, Y. J., Lee, H. C., and Sohng, J. K. (2010). Metabolic engineering of *Streptomyces venezuelae* for malonyl-CoA biosynthesis to enhance heterologous production of polyketides. *Biotechnology Letters*, 32:277–282.

Maksimov, M. O., Pelczer, I., and Link, A. J. (2012). Precursor-centric genome-mining approach for lasso peptide discovery. *Proceedings of the National Academy of Sciences*, 109:15223–15228.

Marahiel, M. and Essen, L.-O. (2009). Nonribosomal peptide synthetases: mechanistic and structural aspects of essential domains. *Methods in Enzymology*, 458:337–351.

Marchler-Bauer, A., Anderson, J. B., Chitsaz, F., Derbyshire, M. K., DeWeese-Scott, C., Fong, J. H., Geer, L. Y., Geer, R. C., Gonzales, N. R., Gwadz, M., et al. (2009). CDD: specific functional annotation with the Conserved Domain Database. *Nucleic Acids Research*, 37:D205–D210.

Marchler-Bauer, A. and Bryant, S. H. (2004). CD-Search: protein domain annotations on the fly. *Nucleic Acids Research*, 32:W327–W331.

Marchler-Bauer, A., Derbyshire, M. K., Gonzales, N. R., Lu, S., Chitsaz, F., Geer, L. Y., Geer, R. C., He, J., Gwadz, M., Hurwitz, D. I., et al. (2014). CDD: NCBI's conserved domain database. *Nucleic Acids Research*, page gku1221.

Marchler-Bauer, A., Lu, S., Anderson, J. B., Chitsaz, F., Derbyshire, M. K., DeWeese-Scott, C., Fong, J. H., Geer, L. Y., Geer, R. C., Gonzales, N. R., et al. (2011). CDD: a Conserved Domain Database for the functional annotation of proteins. *Nucleic Acids Research*, 39:D225–D229.

Margesin, R. and Schinner, F. (2001). Potential of halotolerant and halophilic microorganisms for biotechnology. *Extremophiles*, 5:73–83.

McMurry, J. and Begley, T. P. (2005). *The organic chemistry of biological pathways*. Roberts and Company Publishers.

Medema, M. H., Blin, K., Cimermancic, P., de Jager, V., Zakrzewski, P., Fischbach, M. A., Weber, T., Takano, E., and Breitling, R. (2011a). antiSMASH: rapid identification, annotation and analysis of secondary metabolite biosynthesis gene clusters in bacterial and fungal genome sequences. *Nucleic Acids Research*, 39:W339–W346.

Medema, M. H., Breitling, R., and Takano, E. (2011b). Synthetic biology in Streptomyces bacteria. *Methods Enzymol*, 497:485–502.

Medema, M. H., Trefzer, A., Kovalchuk, A., van den Berg, M., Müller, U., Heijne, W., Wu, L., Alam, M. T., Ronning, C. M., Nierman, W. C., et al. (2010). The sequence of a 1.8-Mb bacterial linear plasmid reveals a rich evolutionary reservoir of secondary metabolic pathways. *Genome Biology and Evolution*, 2:212–224.

Menéndez, N., Nur-e Alam, M., Braña, A. F., Rohr, J., Salas, J. A., and Méndez, C. (2004). Biosynthesis of the antitumor chromomycin A 3 in *Streptomyces griseus*: analysis of the gene cluster and rational design of novel chromomycin analogs. *Chemistry & Biology*, 11:21–32.

Muth, G., Nußbaumer, B., Wohlleben, W., and Pühler, A. (1989). A vector system with temperature-sensitive replication for gene disruption and mutational cloning in streptomycetes. *Molecular and General Genetics MGG*, 219:341–348.

Nguyen, T., Ishida, K., Jenke-Kodama, H., Dittmann, E., Gurgui, C., Hochmuth, T., Taudien, S., Platzer, M., Hertweck, C., and Piel, J. (2008). Exploiting the mosaic structure of trans-acyltransferase polyketide synthases for natural product discovery and pathway dissection. *Nature Biotechnology*, 26:225–233.

Oh, Y.-K., Palsson, B. O., Park, S. M., Schilling, C. H., and Mahadevan, R. (2007). Genome-scale reconstruction of metabolic network in *Bacillus subtilis* based on high-throughput phenotyping and gene essentiality data. *Journal of Biological Chemistry*, 282(39):28791–28799.

Ohnishi, Y., Ishikawa, J., Hara, H., Suzuki, H., Ikenoya, M., Ikeda, H., Yamashita, A., Hattori, M., and Horinouchi, S. (2008). Genome sequence of the streptomycin-producing microorganism *Streptomyces griseus* IFO 13350. *Journal of Bacteriology*, 190:4050–4060.

Okoro, C. K., Brown, R., Jones, A. L., Andrews, B. A., Asenjo, J. A., Goodfellow, M., and Bull, A. T. (2009). Diversity of culturable actinomycetes in hyper-arid soils of the Atacama Desert, Chile. *Antonie van Leeuwenhoek*, 95:121–133.

Olano, C., Lombó, F., Méndez, C., and Salas, J. A. (2008). Improving production of bioactive secondary metabolites in actinomycetes by metabolic engineering. *Metabolic engineering*, 10:281–292.

Ōmura, S., Ikeda, H., Ishikawa, J., Hanamoto, A., Takahashi, C., Shinose, M., Takahashi, Y., Horikawa, H., Nakazawa, H., Osonoe, T., et al. (2001). Genome sequence of an industrial microorganism *Streptomyces avermitilis*: deducing the ability of producing secondary metabolites. *Proceedings of the National Academy of Sciences*, 98:12215–12220.

Orth, J. D., Conrad, T. M., Na, J., Lerman, J. A., Nam, H., Feist, A. M., and Palsson, B. Ø. (2011). A comprehensive genome-scale reconstruction of *Escherichia coli* metabolism—2011. *Molecular Systems Biology*, 7:535.

Orth, J. D., Thiele, I., and Palsson, B. Ø. (2010). What is flux balance analysis? *Nature Biotechnology*, 28:245–248.

Overbeek, R., Olson, R., Pusch, G. D., Olsen, G. J., Davis, J. J., Disz, T., Edwards, R. A., Gerdes, S., Parrello, B., Shukla, M., et al. (2014). The SEED and the Rapid Annotation of microbial genomes using Subsystems Technology (RAST). *Nucleic Acids Research*, 42:D206–D214.

Palaniappan, N., Ayers, S., Gupta, S., Habib, E.-S., and Reynolds, K. A. (2006). Production of hygromycin A analogs in *Streptomyces hygroscopicus* NRRL 2388 through identification and manipulation of the biosynthetic gene cluster. *Chemistry & Biology*, 13:753–764.

Pan, S. J., Rajniak, J., Cheung, W. L., and Link, A. J. (2012). Construction of a single polypeptide that matures and exports the lasso peptide microcin J25. *ChemBioChem*, 13:367–370.

Patnaik, R. and Liao, J. C. (1994). Engineering of *Escherichia coli* central metabolism for aromatic metabolite production with near theoretical yield. *Applied and Environmental Microbiology*, 60:3903–3908.

Pearson, L. A., Hisbergues, M., Börner, T., Dittmann, E., and Neilan, B. A. (2004). Inactivation of an ABC transporter gene, mcyH, results in loss of microcystin production in the cyanobacterium *Microcystis aeruginosa* PCC 7806. *Applied and environmental microbiology*, 70:6370–6378.

Peters, P., Galinski, E., and Trüper, H. (1990). The biosynthesis of ectoine. *FEMS Microbiology Letters*, 71:157–162.

Piel, J. (2010). Biosynthesis of polyketides by *trans*-AT polyketide synthases. *Natural Product Reports*, 27:996–1047.

Pluskal, T., Castillo, S., Villar-Briones, A., and Orešič, M. (2010). MZmine 2: modular framework for processing, visualizing, and analyzing mass spectrometry-based molecular profile data. *BMC Bioinformatics*, 11:1.

Pridham, T. and Gottlieb, D. (1948). The utilization of carbon compounds by some Actinomycetales as an aid for species determination. *Journal of Bacteriology*, 56:107.

Ramos, J. L., Martínez-Bueno, M., Molina-Henares, A. J., Terán, W., Watanabe, K., Zhang, X., Gallegos, M. T., Brennan, R., and Tobes, R. (2005). The TetR family of transcriptional repressors. *Microbiology and Molecular Biology Reviews*, 69:326–356.

Rateb, M. E., Houssen, W. E., Arnold, M., Abdelrahman, M. H., Deng, H., Harrison, W. T., Okoro, C. K., Asenjo, J. A., Andrews, B. A., Ferguson, G., et al. (2011a). Chaxamycins A–D, bioactive ansamycins from a hyper-arid desert *Streptomyces* sp. *Journal of Natural Products*, 74:1491–1499.

Rateb, M. E., Houssen, W. E., Harrison, W. T., Deng, H., Okoro, C. K., Asenjo, J. A., Andrews, B. A., Bull, A. T., Goodfellow, M., Ebel, R., et al. (2011b). Diverse metabolic profiles of a *Streptomyces* strain isolated from a hyper-arid environment. *Journal of Natural Products*, 74:1965–1971.

Rausch, C., Weber, T., Kohlbacher, O., Wohlleben, W., and Huson, D. H. (2005). Specificity prediction of adenylation domains in nonribosomal peptide synthetases (NRPS) using transductive support vector machines (TSVMs). *Nucleic Acids Research*, 33:5799–5808.

Ren, Q., Chen, K., and Paulsen, I. T. (2007). TransportDB: a comprehensive database resource for cytoplasmic membrane transport systems and outer membrane channels. *Nucleic Acids Research*, 35:D274–D279.

Rodríguez-García, A., Santamarta, I., Pérez-Redondo, R., Martín, J. F., and Liras, P. (2006). Characterization of a two-gene operon epeRA involved in multidrug resistance in *Streptomyces clavuligerus*. *Research in Microbiology*, 157:559–568.

Röttig, M., Medema, M. H., Blin, K., Weber, T., Rausch, C., and Kohlbacher, O. (2011). NRPSpredictor2—a web server for predicting NRPS adenylation domain specificity. *Nucleic Acids Research*, page gkr323.

Roullier, C., Guitton, Y., Valery, M., Amand, S., Prado, S., Robiou du Pont, T., Grovel, O., and Pouchus, Y. F. (2016). Automated detection of natural halogenated compounds from LC-MS profiles–Application to the isolation of bioactive chlorinated compounds from marine-derived fungi. *Analytical Chemistry*, 88:9143–9150.

Ryu, Y.-G., Butler, M. J., Chater, K. F., and Lee, K. J. (2006). Engineering of primary carbohydrate metabolism for increased production of actinorhodin in *Streptomyces coelicolor*. *Applied and Environmental Microbiology*, 72:7132–7139.

Sambrook, J., Fritsch, E. F., Maniatis, T., et al. (1989). *Molecular cloning*, volume 2. Cold spring harbor laboratory press New York.

Sanger, F. and Coulson, A. R. (1975). A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *Journal of Molecular Biology*, 94:441–448.

Schupp, T., Toupet, C., and Divers, M. (1988). Cloning and expression of two genes of *Streptomyces pilosus* involved in the biosynthesis of the siderophore desferrioxamine B. *Gene*, 64:179–188.

Segre, D., Vitkup, D., and Church, G. M. (2002). Analysis of optimality in natural and perturbed metabolic networks. *Proceedings of the National Academy of Sciences*, 99:15112–15117.

Shao, Z. H., Ren, S. X., Liu, X. Q., Xu, J., Yan, H., Zhao, G. P., and Wang, J. (2015). A preliminary study of the mechanism of nitrate-stimulated remarkable increase of rifamycin production in *Amycolatopsis mediterranei* U32 by RNA-seq. *Microbial cell factories*, 14:1.

Shen, B. (2003). Polyketide biosynthesis beyond the type I, II and III polyketide synthase paradigms. *Current Opinion in Chemical Biology*, 7:285–295.

Shima, J., Hesketh, A., Okamoto, S., Kawamoto, S., and Ochi, K. (1996). Induction of actinorhodin production by rpsL (encoding ribosomal protein S12) mutations that confer streptomycin resistance in *Streptomyces lividans* and *Streptomyces coelicolor* A3 (2). *Journal of Bacteriology*, 178:7276–7284.

Shirling, E. t. and Gottlieb, D. (1966). Methods for characterization of *Streptomyces* sp.cies1. *International Journal of Systematic and Evolutionary Microbiology*, 16:313–340.

Singh, S. B. and Barrett, J. F. (2006). Empirical antibacterial drug discovery—foundation in natural products. *Biochemical Pharmacology*, 71:1006–1015.

Skoblilova, N., Zakharova, O., Suetin, S., Naumova, I., and Agre, N. (1982). Cell wall composition of *Streptomyces roseoflavus* var. roseofungini and its *Nocardia*-like variant. *Mikrobiologiia*, 52:597–604.

Stachelhaus, T., Mootz, H. D., and Marahiel, M. A. (1999). The specificity-conferring code of adenylation domains in nonribosomal peptide synthetases. *Chemistry & Biology*, 6:493–505.

Streshinskaia, G., Naumova, I., and Panina, L. (1978). Chemical composition of the cell wall of *Streptomyces chrysomallus* which produces the antibiotic aurantin. *Mikrobiologiia*, 48:814–819.

Swiatek, M. A., Tenconi, E., Rigali, S., and van Wezel, G. P. (2012). Functional analysis of the N-acetylglucosamine metabolic genes of *Streptomyces coelicolor* and role in control of development and antibiotic production. *Journal of Bacteriology*, 194:1136–1144.

Tahlan, K., Ahn, S. K., Sing, A., Bodnaruk, T. D., Willems, A. R., Davidson, A. R., and Nodwell, J. R. (2007). Initiation of actinorhodin export in *Streptomyces coelicolor*. *Molecular microbiology*, 63:951–961.

Tahlan, K., Yu, Z., Xu, Y., Davidson, A. R., and Nodwell, J. R. (2008). Ligand recognition by ActR, a TetR-like regulator of actinorhodin export. *Journal of molecular biology*, 383:753–761.

Tang, G.-L., Cheng, Y.-Q., and Shen, B. (2004). Leinamycin biosynthesis revealing unprecedented architectural complexity for a hybrid polyketide synthase and nonribosomal peptide synthetase. *Chemistry & Biology*, 11:33–45.

Tang, G.-L., Cheng, Y.-Q., and Shen, B. (2006). Polyketide chain skipping mechanism in the biosynthesis of the hybrid nonribosomal peptide-polyketide antitumor antibiotic leinamycin in *Streptomyces atroolivaceus* S-140. *Journal of Natural Products*, 69:387–393.

Thanapipatsiri, A., Gomez-Escribano, J. P., Song, L., Bibb, M. J., Al-Bassam, M., Chandra, G., Thamchaipenet, A., Challis, G. L., and Bibb, M. J. (2016). Discovery of Unusual Biaryl Polyketides by Activation of a Silent *Streptomyces venezuelae* Biosynthetic Gene Cluster. *ChemBioChem*, 17:2189–2198.

Um, S., Kim, Y.-J., Kwon, H., Wen, H., Kim, S.-H., Kwon, H. C., Park, S., Shin, J., and Oh, D.-C. (2013). Sungsanpin, a lasso peptide from a deep-sea streptomycete. *Journal of Natural Products*, 76:873–879.

Vaillancourt, F. H., Yeh, E., Vosburg, D. A., O'Connor, S. E., and Walsh, C. T. (2005). Cryptic chlorination by a non-haem iron enzyme during cyclopropyl amino acid biosynthesis. *Nature*, 436:1191–1194.

Van Domselaar, G. H., Stothard, P., Shrivastava, S., Cruz, J. A., Guo, A., Dong, X., Lu, P., Szafron, D., Greiner, R., and Wishart, D. S. (2005). BASys: a web server for automated bacterial genome annotation. *Nucleic Acids Research*, 33:W455–W459.

Walsh, C. and Wright, G. (2005). Introduction: antibiotic resistance. *Chemical Reviews*, 105:391–394.

Walsh, C. T., Chen, H., Keating, T. A., Hubbard, B. K., Losey, H. C., Luo, L., Marshall, C. G., Miller, D. A., and Patel, H. M. (2001). Tailoring enzymes that modify nonribosomal peptides during and after chain elongation on NRPS assembly lines. *Current Opinion in Chemical Biology*, 5:525–534.

Weber, T., Blin, K., Duddela, S., Krug, D., Kim, H. U., Bruccoleri, R., Lee, S. Y., Fischbach, M. A., Müller, R., Wohlleben, W., et al. (2015). antiSMASH 3.0—a comprehensive resource for the genome mining of biosynthetic gene clusters. *Nucleic Acids Research*, 43:W237–W243.

Wilkinson, B. and Micklefield, J. (2007). Mining and engineering natural-product biosynthetic pathways. *Nature Chemical Biology*, 3:379–386.

Zaretskaia, M. and Polin, A. (1987). Polysaccharide composition of the cell wall of *Streptomyces antibioticus* RIA-594 (39), a producer of the antibiotic oleandomycin. *Antibiotiki i Meditsinskaia Biotekhnologiia = Antibiotics and Medical Biotechnology*, 32:529–533.

Zha, W., Rubin-Pitel, S. B., Shao, Z., and Zhao, H. (2009). Improving cellular malonyl-CoA level in *Escherichia coli* via metabolic engineering. *Metabolic Engineering*, 11:192–198.

Zhao, B., Lin, X., Lei, L., Lamb, D. C., Kelly, S. L., Waterman, M. R., and Cane, D. E. (2008). Biosynthesis of the sesquiterpene antibiotic albaflavenone in *Streptomyces coelicolor* A3 (2). *Journal of Biological Chemistry*, 283:8183–8189.

Zimmermann, M., Hegemann, J. D., Xie, X., and Marahiel, M. A. (2013). The astexin-1 lasso peptides: biosynthesis, stability, and structural studies. *Chemistry & Biology*, 20:558–569.

Appendix A

Strains and plasmids

Table A.1: List of strains used in this work.

Strain	Description	Reference or source
E. coli DH5α	Strain used for routine cloning. F- Φ80*lac*ZΔM15 Δ(*lac*ZYA-*arg*F) U169 *rec*A1 *end*A1 *hsd*R17(r_{k-}. m_{k+}) *pho*A *sup*E44 *thi*-1 *gyr*A96 *rel*A1 λ-	Grant et al., 1990
E. coli TOP10	Strain used for routine cloning. F- mcrA Δ(mrr-hsdRMS-mcrBC) Φ80lacZΔM15 ΔlacX74 nupG recA1 araD139 Δ(ara-leu)7697 galE15 galK16 rpsL(StrR) endA1 λ-	
E. coli DH10B	Strain used for routine cloning. F- endA1 recA1 galE15 galK16 nupG rpsL ΔlacX74 Φ80lacZΔM15 araD139 Δ(ara-leu)7697 mcrA Δ(mrr-hsdRMS-mcrBC) λ-	Grant et al., 1990
E. coli ET12567/pUZ8002	Methylation deficient strain used for conjugation with *Streptomyces*. pUZ8002 provides conjugation machinery. dam13::Tn9 (Chloramphenicol resistance) dcm-6 hsdM hsdR recF143 zjj-201::Tn10 galK2 galT22 ara14 lacY1 xyl-5 leuB6 thi-1 tonA31 rpsL136 hisG4 tsx-78 mtlI glnV44. pUZ8002 (kanamycin resistance)	MacNeil et al., 1992; pUZ8002, J. Wilson and D. Figurski, unpublished
E. coli TOP10/pR9604	Strain used for routine cloning carrying conjugation plasmid pR9406 (carbenicillin resistance)	pR9406. A. Siddique and D. Figurski. unpublished
S. leeuwenhoekii	Wild type strain	Busarakam et al., 2014

Continuation of Table A.1

Strain	Description	Reference or source
S. coelicolor M1152	*S. coelicolor* M145. Δact Δred Δcpk Δcda rpoB[C1298T]	Gomez-Escribano and Bibb, 2011
S. coelicolor M1154	*S. coelicolor* M145. Δact Δred Δcpk Δcda rpoB[C1298T] rpsL[A262G]	Gomez-Escribano and Bibb, 2011
S. albus J1074		
S. leeuwenhoekii M1653	Δ*cxmK::neo*	Castro et al., 2015
S. leeuwenhoekii M1600	Δhalogenase::neo	This work
S. leeuwenhoekii M1601	ΔTransAT::neo	This work
S. leeuwenhoekii M1602	Δhalogenase::neo::halogenase::hyg	This work
S. leeuwenhoekii M1603	ΔTransAT::neo::TransAT::hyg	This work
S. leeuwenhoekii M1604	Halogenase::hyg	This work
S. leeuwenhoekii M1605	LysR:hyg	This work
S. coelicolor M1606	*S. coelicolor* M1152 carrying PAC 18O	This work
S. coelicolor M1607	*S. coelicolor* M1152 carrying PAC 14B	This work
S. coelicolor M1608	*S. coelicolor* M1152 carrying PAC 14B::LysR::hyg	This work
S. coelicolor M1609	*S. coelicolor* M1154 carrying PAC 18O	This work
S. coelicolor M1610	*S. coelicolor* M1154 carrying PAC 14B	This work
S. coelicolor M1611	*S. coelicolor* M1154 carrying PAC 18O::LysR::hyg	This work
S. albus M1612	*S. albus* J1074 carrying PAC 14B	This work
S. albus M1613	*S. albus* J1074 carrying PAC 14B::LysR::hyg	This work

Continuation of Table A.1

Strain	Description	Reference or source
S. leeuwenhoekii M1614	ΔChaxamycin gene cluster	This work
S. leeuwenhoekii M1615	ΔTetR::neo	This work
S. leeuwenhoekii M1616	Sle9280::hyg	This work
S. leeuwenhoekii M1617	*S. leeuwenhoekii* M1614 with ΔTetR::neo	This work
S. leeuwenhoekii M1618	*S. leeuwenhoekii* M1614 carrying Sle9280::hyg	This work
S. leeuwenhoekii M1619	*S. leeuwenhoekii* M1614 ΔTetR	This work

Table A.2: List of plasmids used in this work.

Plasmid	Description	Reference or source
pBluescript II SK(+)	General cloning vector, *amp*R (ampicillin resistance)	(Alting-Mees and Short, 1989)
pTC192-Km	Source of *neo* (kanamycin resistance gene)	(Rodríguez-García et al., 2006)
pGM1190	Contains temperature-sensitive replication origin of pSG5, self-replicative, temperature sensitive, *tsr*, *aac*(3)IV (apramycin resistance). *ori*T, to terminator, PtipA, RBS, *fd* terminator	pSG5 (Muth et al., 1989); pGM1190, G. Muth, unpublished
pESAC13	PAC vector (P1-phage replicon) for genomic library construction; conjugative (*ori*T from RK2), integrative (ΦC31 attP), *tsr* (thiostrepton resistance in Streptomyces), *neo* (kanamycin resistance in *E. coli*), P1 rep, *sac*BII	Sosio et al., 2000; pESAC13, M. Sosio, unpublished
pIJ10257	Expression vector for *Streptomyces* with *erm*E* promoter, *hyg* (hygromycin B resistance), conjugative (*ori*T from RK2), integrates into ΦBT1 attachment site *attP*	(Hong et al., 2005)

Continuation of Table A.2

Plasmid	Description	Reference or source
pIJ12801	Derivative of pGM1190 with VR007/VR008-*neo*-VR009/VR010 fragment for deletion of Sle09470 in *S. leeuwenhoekii*	This work
pIJ12803	Derivative of pGM1190 with VR003/VR004-*neo*-VR005/VR006 fragment for deletion of Sle09500 in *S. leeuwenhoekii*	This work
pIJ12804	Derivative of pGM1190 with VR007/VR008-VR009/VR010 fragment for deletion of Sle09470 in *S. leeuwenhoekii*	This work
pIJ12805	Derivative of pGM1190 with VR003/VR004-VR005/VR006 fragment for deletion of Sle09500 in *S. leeuwenhoekii*	This work
pIJ12806	Derivative of pESAC13 with 18O insert	This work
pIJ12808	Derivative of pESAC13 with 14B insert	This work
pIJ12809	Derivative of pGM1190 with VJ001/VJ002-VJ003/VJ004 fragment for the deletion of chaxamycin gene cluster in *S. leeuwenhoekii* M1653	This work
pIJ12810	Derivative of pIJ10257 with VR039/VR044 fragment for Sle09760 (*lys*R) overexpression in *S. leeuwenhoekii*	This work
pIJ12811	Derivative of pIJ10257 with VR029/VR030 fragment for Sle09470 overexpression in *S. leeuwenhoekii*	This work
pIJ12812	Derivative of pIJ10257 with VR027/VR028 fragment for Sle09500 overexpression in *S. leeuwenhoekii*	This work
pIJ12813	Derivative of pGM1190 with VR035/VR036-*neo*-VR037/VR038 for deletion of Sle09560 (*tet*R) in *S. leeuwenhoekii* and in *S. leeuwenhoekii* M1614	This work
pIJ12814	Derivative of pGM1190 with VR035/VR036-VR037/VR038 for deletion of Sle09560 (*tet*R) in *S. leeuwenhoekii* and in *S. leeuwenhoekii* M1614	This work
pIJ12817	Derivative of pIJ10257 with VR050/VR051 fragment for Sle09280 overexpression in *S. leeuwenhoekii* and in *S. leeuwenhoekii* M1614	This work

Appendix B

List of primers and maps of vectors used in this work

B.1 List of primers

Table B.1: List of primers used in this work.

Name	Primer sequence 5' → 3'	Notes
VR003	aaaagcttaacctgatgatgcgccagaa	Amplifies a 1.9 Kb region upstream of *sle09500* gene. Includes restriction sites for HindIII and XbaI, respectively.
VR004	aatctagactgtccgggaaagagccag	
VR005	aatctagagaccgcctacgccaactg	Amplifies a 1.7 Kb region downstream of *sle09500* gene. Includes restriction sites for XbaI and EcoRV, respectively.
VR006	aagatatcgagaagtcgaagctcaccga	
VR007	aaaagcttgcgtgctgggcatcctca	Amplifies a 1.7 Kb region upstream of *sle09470* gene. Includes restriction sites for HindIII and XbaI, respectively.
VR008	tttctagacgtgtacgcggttctctcc	
VR009	aatctagaggacaggacgcctacggg	Amplifies a 1.7 Kb region downstream of *sle09470* gene. Includes restriction sites for XbaI and EcoRI, respectively.
VR010	aagaattccgaactcccggtacagctg	
VR016	cgtgatcggcttctcctacc	Used to amplify a region of 0.47 Kb upstream of the hybrid *trans*-AT PKS/NRPS BGC.
VR017	gaaggaggcgaccgtgac	
VR018	acaggaactccttggcatcg	Used to amplify a region of 0.45 Kb downstream of the hybrid *trans*-AT PKS/NRPS BGC.
VR019	catcgtcgtaggttcggaca	
VR035	ttgaattcgcgaccgtcccagaactg	Amplifies a 1.5 Kb region upstream of *sle09560* (*tet*R) gene. Includes restriction sites for EcoRI and XbaI, respectively.
VR036	aatctagacgtacgcgtggacatggaatcc	

Continuation of Table B.1

Name	Primer sequence 5' → 3'	Notes
VR037	tctagaagggagcagtacgaggagtc	Amplifies a 1.6 Kb region downstream of *sle09560* (*tet*R) gene. Includes restriction sites for XbaI and NdeI, respectively.
VR038	ggctggacgtccgacttccatatg	
VR040	aacatatgaagaaggcgtacgagg	Amplifies a 4.7 Kb region corresponding to the Lasso-peptide 1 BGC. Includes restriction sites for NdeI and HindIII, respectively.
VR041	gaaagctttcgcgtccccggaccacg	
VR050	tcgtcatatcatatggatgcg	Amplifies *sle09280*. Includes restriction sites for NdeI and HindIII, respectively.
VR051	aaaagcttgtgcacacgggaacggaca	
VR052	caggaagtgacccgtccgtgaa	Used for confirmation of *sle09560* (*tet*R) deletion.
VR053	accatcaggtcggccagacaga	
VR054	cttggtctcgatccattggt	Used for confirmation of Δchaxamycin BGC. Amplifies a 1.6 Kb region.
VR055	acatcgtcccgagtgagaag	
VJ001	ttgaattcgtcggctactacgtccgc	Amplifies a 2.5 Kb from the right region of the chaxamycin BGC. Includes restriction sites for EcoRI and XbaI, respectively.
VJ002	tttctagactcgcacaggattcctcgg	
VJ003	tttctagagtgggtcaccgggaagtg	Amplifies a 2.5 Kb from the left region of the chaxamycin BGC. Includes restriction sites for XbaI and NdeI, respectively.
VJ004	ttcatatgcttgaggtggcactgatcgg	

Appendix C

Buffers, solutions and culture media

C.1 Buffer solutions

C.1.1 Buffer SET

For preparing 50 ml of buffer SET the following volumes were used:

NaCl (0.9 % w/v)	3.75 ml
EDTA (0.25 M pH 8)	5 ml
Tris-HCl (1 M pH 8)	1 ml
dH_2O	40.25 ml

The final concentration of each component is: 75 mM NaCl; 25 mM EDTA; 20 mM Tris-HCl.

C.1.2 Buffer STET

For preparing 50 ml of buffer STET the following volumes were used:

Sucrose (20 % w/v)	20 ml
Triton X-100 (10 % v/v)	2.50 ml
EDTA (0.25 M pH 8)	10 ml
Tris-HCl (1 M pH 8)	2.50 ml
dH_2O	15 ml

The final concentration of each component is: Sucrose (8 % w/v); Triton X-100 (0.5 % v/v); 50 mM EDTA; 50 mM Tris-HCl.

C.1.3 Buffer TE

For preparing 100 ml of buffer TE the following volumes were used:

EDTA (0.25 M pH 8) ..... 0.40 ml
Tris-HCl (1 M pH 8) ..... 1 ml
dH_2O ..... 98.6 ml

The final concentration of each component is: 1.0 mM EDTA; 10 mM Tris-HCl.

C.2 Microbiology solutions

All the microbiology solutions used are listed in Table C.1. The stock solution were prepared by dissolving the compound in the indicated solvent, afterwards the solutions were sterilized using a 0.2 μm filter and stored at -20 °C.

Table C.1: Microbiology solutions

Solution	**Catalogue number**	**Dissolved in**	**Stock concentration**	**Working concentration**
Apramycin	Apramycin sulphate Sigma cat. no. A2024	dH_2O	50 mg/ml	50 μg/ml
Carbenicillin	Carbenicillin disodium salt Invitrogen cat. no. 10177-012	dH_2O	50 mg/ml	50 μg/ml
Chloramphenicol	Chloramphenicol Calbiochem cat. no. 220551	Absolute ethanol	50 mg/ml	50 μg/ml
Hygromycin B*	Hygromycin B from *Streptomyces hygroscopicus* Sigma cat. no. 238813	PBS	50 mg/ml	80 μg/ml (*E. coli*); 40 μg/ml (*Streptomyces*)
Kanamycin	Kanamycin Gibco cat. no. 11815-024	dH_2O	50 mg/ml	50 μg/ml
Nalidixic acid	Nalidixic acid Sigma cat. no. N8878	0.3 M NaOH	25 mg/ml	20 μg/ml
Thiostrepton	Thiostrepton Sigma cat. no. T8902	DMSO	50 mg/ml	50 μg/ml
IPTG	IPTG Fermentas cat. no. R0392	dH_2O	0.1 M	0.1 mM
Lyzosyme	Lysozyme from chicken egg Sigma cat. no. L6876	dH_2O	50 mg/ml	50 μg/ml
X-Gal*	X-Gal Fermentas cat. no. R0404	DMSO	20 mg/ml	20 μg/ml

*Light sensitive

C.3 Culture medium

All media were autoclaved at 121 °C for 20 minutes. If needed, antibiotics or other solutions (like $MgCl_2$) were added after autoclaving.

C.3.1 Agar media

The components of DNA medium are (Kieser et al., 2000):

Difco Nutrient Agar	4.6 g
dH_2O	200 ml

The components of LB agar are (Sambrook et al., 1989):

Yeast extract	5 g
Tryptone	10 g
NaCl	10 g
Agar	15 g
dH_2O	1000 ml

The components of SFM are (Hobbs et al., 1989):

Soya flour	4.0 g
Mannitol	4.0 g
Agar	4.0 g
dH_2O	200 ml

The components of MM are (Pridham and Gottlieb, 1948):

$(NH_4)_2SO_4$	2.64 g
KH_2PO_4	2.38 g
$K_2HPO_4 \cdot 3H_2O$	5.65 g
$MgSO_4 \cdot 7H_2O$	1.0 g
Pridham and Gottlieb trace salts*	1.0 ml
Agar	15.0 g
dH_2O	1000 ml

Pridham and Gottlieb trace salts composition*

$CuSO_4 \cdot 5H_2O$ ..... 0.64 g
$FeSO_4 \cdot 7H_2O$ ..... 0.11 g
$MnCl_2 \cdot 4H_2O$ ..... 0.79 g
$ZnSO_4 \cdot 7H_2O$ ..... 0.15 g
dH_2O ..... 100 ml

The pH was adjusted to 6.8-7.0 before autoclaving. This media was used to test different carbon sources, each one was added at a final concentration of 1 % w/v.

C.3.2 Liquid media

The components of LB medium are (Sambrook et al., 1989):

Yeast extract ..... 5.0 g
Tryptone ..... 10 g
NaCl ..... 10 g
dH_2O ..... 1000 ml

The components of ISP2 medium are (Shirling and Gottlieb, 1966):

Yeast extract ..... 4.0 g
Glucose ..... 4.0 g
Malt extract ..... 10.0 g
dH_2O ..... 1000 ml

In the case of mISP2 glucose is replaced by 10.0 g of glycerol. In both cases, the pH was adjusted to 7.2 before autoclaving.

The components of YEME medium are (Kieser et al., 2000):

Difco yeast extract ..... 3 g
Difco Bacto-peptone ..... 5 g
Oxoid malt extract ..... 3 g
Glucose ..... 10 g
Sucrose ..... 340 g
dH_2O ..... 1000 ml

After autoclaving add:

$MgCl_2 \cdot 6H_2O$ (2.5 M) ..... 2 ml (5 mM final concentration)

The components of TSB medium are (Kieser et al., 2000):

Tryptone Soya Broth powder	30 g
dH_2O	1000 ml

The components of 2x YT medium are (Kieser et al., 2000):

Difco Bacto tryptone	16 g
Difco Bacto yeast extract	10 g
NaCl	5 g
dH_2O	1000 ml

The components of SOC medium are (Hanahan, 1983):

Trypthone	20 g
Yeast extract	5 g
Glucose	3.603 g
$MgSO_4 \cdot H_2O$	2.467 g
$MgCl_2 \cdot 6H_2O$	2.033 g
NaCl	0.584 g
KCl	0.186 g
dH_2O	1000 ml

The components of R3 medium are (Shima et al., 1996):

Glucose	10 g
Yeast extract	5 g
L-proline	3 g
Casaminoacids	0.1 g
$MgCl_2 \cdot 6H_2O$	10 g
$CaCl_2 \cdot 2H_2O$	4 g
K_2SO_4	0.2 g
KH_2PO_4	0.05 g
TES	5.6 g
dH_2O	1000 ml

Add 1 ml per litre of the following trace elements solution (Kieser et al., 2000):

$ZnCl_2$ ..... 0.04 g
$FeCl_3 \cdot 6H_2O$ ..... 0.2 g
$CuCl_2 \cdot 2H_2O$ ..... 0.01 g
$MnCl_2 \cdot 4H_2O$ ..... 0.01 g
$Na_2B_4O_7 \cdot 10H_2O$ ..... 0.01 g
$(NH_4)_6Mo_7O_{24} \cdot 4H_2O$ ..... 0.01 g
dH_2O ..... 1000 ml

The pH of R3 medium was adjusted to 7.2 before autoclaving.

The components of DM medium are:

Glycerol ..... 10 g
L-Monosodium glutamate monohydrate ..... 5.0 g
Inositol ..... 0.4 g
NaCl ..... 2.3 g
$CaCO_3$ ..... 0.25 g
KH_2SO_4 ..... 2 g
$(NH_4)_2SO_4$ ..... 1.5 g
$MgSO_4 \cdot H_2O$ ..... 0.5 g
$FeSO_4 \cdot 7H_2O$ ..... 0.025 g
$CoCl_2$ ..... 0.01 g
$ZnSO_4 \cdot 7H_2O$ ..... 0.01 g
dH_2O ..... 1000 ml

The mDM medium is the same as DM with the addition of 0.01 g/L of vitamin B12.

The components of mLPM (based on the formulation of leinamycin production media (Cheng et al., 2002)) medium are:

Soluble starch ..... 40 g
Soya flour ..... 5 g
$CaCO_3$ ..... 5 g
Corn flour ..... 2.5 g
KH_2SO_4 ..... 0.5 g
$MgSO_4 \cdot H_2O$ ..... 0.25 g
Methionine ..... 0.1 g
$CoCl_2$ ..... 0.01 g
$ZnSO_4 \cdot 7H_2O$ ..... 0.04 g
Vitamin B12 ..... 0.01 g
dH_2O ..... 1000 ml

The pH was adjusted to 7.0 before autoclaving.

Appendix D

Protocols

D.1 Boiling plasmid DNA extraction

1. Centrifuge tubes at 5.000 rpm for 5 min.
2. Discard supernantant (without touching the inside of the tube).
3. Resuspend in 200 μl of STET buffer.
4. Add 10 μl of lyzozyme 25 mg/ml in the lid of the tube.
5. Stir the tubes in the vortex.
6. Leave in boiling water for 45 s.
7. Centrifugate at 13.000 rpm for 5 min.
8. Remove the precipitate with a toothpick.
9. Add 250 μl of isopropanol and mix by inversion. DNA will precipitate.
10. Leave 15 min at room temperature.
11. Centrifugate at 13.000 rpm for 20 min. Discard supernatant.
12. Let dry for a while (10 min).
13. Resuspend in 50 μl of H_2O. mix gently.
14. Leave at 4 °C overnight.

D.2 Genomic DNA extraction

1. Five millilitres of a 2-3 days liquid culture of *Streptomyces* in TSB:YEME. Keep in glycerol 20 % if not used immediately.
2. Centrifugate for 10 minutes at 3.500 rpm. Discard supernatant.
3. Add 5 ml of Buffer SET and resuspend the pellet.
4. Add 150 μl of lyzozyme 50 mg/ml. Incubate at 37 °C for 2 h. Shake the sample every half hour.

5. Add 300 μl of SDS 20 %.
6. Add 75 μl of proteinase K and gently shake for 1.5 h at 60 °C until it become transparent.
7. Add 2 ml of 5 M NaCl. mix well and slowly. gDNA will precipitate.
8. Add analytic grade chloroform 15 ml (V_T).
9. Mix by inversion for 30 min in a rotatory wheel.
10. Centrifuge for 20 min at 4.000 rpm at room temperature.
11. Remove the first phase with a sterile chapped tip and put in a new tube (slowly).
12. Add the same amount of isopropanol and gently shake by inversion.
13. Add 1 ml of ethanol 70 % to a clean and sterile eppendorf tube.
14. With the help of a pasteur pipette "fish" out the gDNA and put into the eppendorf tube with 1 ml of ethanol 70 %.
15. Centrifuge for 1 min at 4.000 rpm. Carefully discard supenantant.
16. Resuspend the samples in ethanol 100 % and store at -20 °C. or resuspend in buffer TE.
17. The eppendorf tubes with buffer TE and gDNA could be left at 60 °C for 2 h, for enhanced solubility.

D.3 Chemical transformation of *E. coli*

1. Mix 10 μl of ligation mixture with 100 μl of chemical competent *E. coli* cells.
2. Keep 30 min on ice.
3. Heat shock at 42 °C for 45 s.
4. Keep on ice for 1 min.
5. Add 900 μl of SOC or LB medium.
6. Incubate in shaker at 37 °C for 1 h.
7. Cultivate on LB plates (supplemented with relevant antibiotics).
8. Grow overnight at 37 °C.

D.4 Preparation of *E. coli* electro-competent cells

1. Start an overnight culture of *E. coli* strain in 5 ml of LB at 30 °C/37 °C with (or without) relevant antibiotics and stir at 250 rpm.
2. The next morning, inoculate 50 ml of fresh LB, supplemented with (or without) relevant antibiotics, with 500 μl of *E. coli* from the overnight culture. Incubate at 30 °C/37 °C at 250 rpm until OD_{600nm} is between 0.5–0.7. Place glycerol solution on ice for further use.
3. Place the culture on ice for 10 min.
4. Centrifuge culture in 50 ml conical centrifuge tubes at 4,500 rpm for 5 min at 4 °C.
5. Remove supernatant and resuspend cell pellet in 20 ml of 10% glycerol (kept on ice).
6. Centrifuge at 4,500 rpm for 5 min at 4 °C.
7. Remove supernatant and resuspend cell pellet in 20 ml of 10% glycerol (kept on ice).

8. Centrifuge at 4 °C and 4,500 rpm for 5 min.
9. Remove supernatant and resuspend cell pellet in 1 ml 10% glycerol (kept on ice).
10. Prepare 100 μl aliquots in 1.5 ml sterile Eppendorf tubes and store at -80 °C.

D.5 Transformation of *E. coli* by electroporation

1. Mix 100 μl of electro-competent *E. coli* (see D.4) with 10 μl of purified plasmid or ligation mixture in a 1.5 ml Eppendorf tube. Use cold tips.
2. Place the mixture in a freeze electroporation cuvette (0.2 cm gap width) and carry out electroporation at 2,500 V. Use ice-cold cuvette.
3. Immediately add 900 μl ice-cold LB and incubate for 1 h at 37 °C.
4. Spread volumes of 100 μl, 150 μl and 200 μl on LB plates supplemented with relevant antibiotics.
5. Incubate at 37 °C overnight and screen for the desired transformant.

D.6 Conjugation between *E. coli* and *Streptomyces*.

1. Inoculate a colony of *E. coli* ET12567 pUZ8002 carrying the plasmid of interest into 10 ml LB containing chloramphenicol (50 μ/ml), kanamycin (50 μ/ml) and relevant antibiotic (for selection of the plasmid to be transferred). Grow overnight at 37 °C and 200 rpm.
2. Inoculate 100-200 μl of overnight culture into 10 ml of fresh LB plus relevant antibiotics. Grow at 37 °C and 200 rpm until an OD_{600} of 0.4 is reached (4 h approximated).
3. Centrifuge at 4,500 rpm for 5 min at RT. Remove supernatant.
4. Resuspend with 10 ml of fresh LB.
5. Centrifuge at 4,500 rpm for 5 min at RT. Discard supernatant.
6. Repeat step 4 and 5. Resuspend in 500 μl of LB.
7. Add 20-50 μl of *Streptomyces* spores to 500 μl of 2x YT broth. Heat-shock at 50 °C for 10 min. Allow to cool by leaving at RT.
8. Mix 500 μl of *E. coli* with 500 μl of heat-shocked *Streptomyces* spores. Centrifuge at 5,000 rpm for 3 min. Discard the supernatant and resuspend in 300 μl of LB
9. Plate out 100 μl of the mix in SFM agar with 10 mM $CaCl_2$ and 10 mM $MgCl_2$ or 60 $CaCl_2$. Incubate at 30 °C for 18-22 h.
10. Overlay the plate with 1 ml of water with nalidixic acid plus relevant antibiotics (see Table D.1 for volumes used). Distribute the solution evenly on the plate and let dry. Continue incubation at 30 °C.
11. After 2-4 days select ex-conjugants and plate them in SFM agar with nalidixic acid plus relevant antibiotics. Incubate at 30 °C or 37 °C if the transferred plasmid is temperature sensitive.
12. If a gene deletion is intended, grow the ex-conjugants in SFM supplemented with relevant antibiotics to select for double or single crossovers (Figure D.1). Make a spore stock and

plate dilutions of it (enough to have isolated colonies) in SFM plus relevant antibiotics and grow at 30 °C or 37 °C. After 3 to 5 days, perform replica plates to find double crossovers.

Table D.1: Volumes of antibiotics used for overlay the conjugation plates.

Antibiotic	**μl for 1 ml overlay**	**Final concentration (μg/ml)**
Apramycin	25	50
Hygromycin	20	40
Kanamycin	25	50
Nalidixic acid	20	20
Thiostrepton	30	60

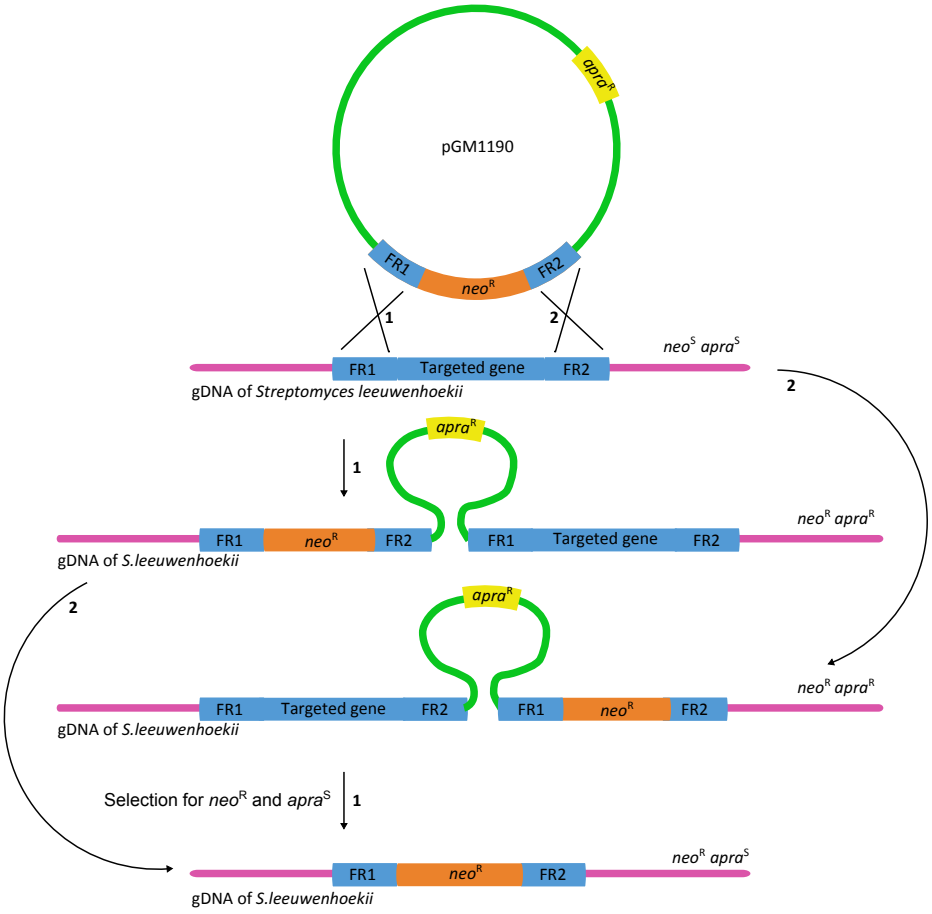


Figure D.1: Generation of a gene deletion in *S. leeuwenhoekii* through conjugation of pGM1190 derivatives and double recombination events.

D.7 Triparental mating using *S. coelicolor* as recipient

.

1. Grow overnight cultures of *E. coli* DH10B (containing the PAC clone, neo^R in *E. coli*, tsr^R in *Streptomyces*), *E. coli* TOP10 (containing the self-transmissible helper plasmid pR9604 amp^R) and *E. coli* ET12567 (recipient Cm^R) into exponential phase.

2. Centrifuge at 4,500 rpm for 5 min at RT. Remove supernatant. Resuspend in 10 ml of LB.
3. Repeat the previous step but resuspend each pellet in 500 μl of LB.
4. Spot 10 to 20 μl of each strain onto the same location of a LB agar plate (without antibiotics). Wait until the drops are dry and incubate at 37 °C.
5. Streak the overnight spots from the LB agar plate onto LB agar plates supplemented with kanamycin, carbenicillin and chloramphenicol. Grow overnight at 37 °C.
6. Select ex-conjugants of *E. coli* ET12567 containing the PAC clone and pR9604, and inoculate them into LB broth (supplemented with kanamycin, carbenicillin and chloramphenicol).
7. Proceed to conjugate the ex-conjugants to *S. coelicolor* M1152 or M1154 following the protocol describe in Section D.6, but plate the conjugation mixture on R2-S agar plates and select the ex-conjugants with thiostrepton and nalidixic acid.
8. When possible, confirm the presence of the PAC clone by PCR.

D.8 Growth conditions for production of specialised metabolites

1. Erlenmeyer flasks of 250 ml were coated with 5 % dimethylchlorosilane (Sylon CTTM, Supelco catalogue number 33065-U) to prevent the mycelium from sticking to the walls. The flasks were allowed to dry before they were autoclaved and used.
2. Glass beads of 3 mm (Merck catalogue number 104015) were added to the flasks to improve the dispersion of the culture.
3. Inoculate the strains of interest into the 250 ml Erlenmeyer flasks with 50 ml of ISP2 or mISP2, incubate for 3-4 days at 30 °C and 200 rpm.
4. Take 5 ml and wash with saline solution. Resuspend in the production media or saline solution, and measure the OD_{600}.
5. Inoculate fresh 250 ml Erlenmeyer flasks with 50 ml of production media with the necessary volume of the seed culture to accomplish a OD_{600} of 0.2. The recipes for the production media are shown in Section C.3.2.
6. Grow for 3-10 days at 30 °C and 200 rpm. If needed, take samples each day.

D.9 Bioassays: testing the antibiotic activity

1. Put an aliquot of *B. subtilis* or *M. luteus* into 10 ml of LB. Grow overnight at 37 °C and 200 rpm.
2. Inoculate 50 ml of fresh LB with 1 ml of the overnight culture. Cultivate at 30 °C and 200 rpm until an OD_{600} of 0.6 is reached.
3. Melt sterile LB agar and let cool to 40-45 °C. Add the necessary volume of the previous culture to obtain an OD_{600} of 0.0125 in the LB agar. Quickly, to prevent early solidification of the agar, mix gently and pour 25 ml onto each plate. To measure the volume a sterile 50 ml falcon tube can be used.

4. Once the LB agar plates are solid, use the back of a tip to make wells in the agar.
5. Put 50 μl of each sample in each well. As positive control use 100 μg/ml of apramycin or 100 μg/ml of carbenicillin, and as negative control use either sterile media or methanol, depending on the type of sample.
6. Carefully, take the plates and leave them at 4 °C for 2-4 hours, to allow diffusion of the liquid.
7. Incubate at 30 °C or at RT for 1-2 days.

D.10 Chromatographic conditions

Three equipments were used during this work. The chromatographic conditions used for each one were as follow:

Shimadzu LC-MS system coupled to LCMS-IT-ToF mass spectrometer. Five μl were injected into the LC-MS system that was equipped with a Kinetex XB-C18 2.6 μm 100 Å 50x2.10 mm column. The elution flux was set to 0.6 ml per min with a gradient of 0.1 % v/v formic acid in water (mobile phase A) and methanol (mobile phase B) as follows: 0 min, 2% B; 1 min, 2 % B; 8 min, 100 % B; 9.3 min, 100 % B; 9.5 min, 2 % B; 11.2 min, 2 % B.

Bruker Daltonik electrospray-ion tramp ESI-IT Esquire 4000. Twenty μl were injected into the LC-MS system that was equipped with a C18 of 150x2.00 mm column. The elution flux was set to 0.3 mL per min with a gradient of 0.1 % v/v formic acid in water (mobile phase A) and methanol (mobile phase B) as follows: 0 min, 2 % B; 6 min, 2 % B; 43.5 min, 100 % B; 51.7 min, 2 % B; 60 min, 2 % B.

Maxis II QTOF. Five μl were injected into the LC-MS system that was equipped with a Kinetex XB-C18 2.6 μm 100 Å 100x2.10 mm column. The elution flux was set to 1.0 ml per min with a gradient of 0.1 % v/v formic acid in water (mobile phase A) and acetonitrile plus formic acid 0.1 % (mobile phase B) as follows: 0 min, 2 % B; 12 min, 100 % B; 15 min, 2 % B.

D.11 Metabolic profile comparison: analysis of the samples

1. The output files of the HPLC MS/MS were transformed to the format mzXML using CompassXport 3.0.9.2
2. The data was analysed using either Mass++ (http://www.masspp.jp/) or Mzmine 2 (Pluskal et al., 2010) software.
3. When using Mzmine the next steps were followed:
   (a) Raw data import.
   (b) Peak detection: using either **targeted peak detection** or **mass detection** followed by chromatogram builder and chromatogram deconvolution. The parameters used were different depending of the sample information.
   (c) Isotopic peak grouper for removing isotopes.

(d) Join aligner for alignment of the peaks of different samples.

(e) Study of the identified peaks.

Appendix E

Supplementary information for Chapter One

E.1 Draws of the BGC

E.1.1 Code used to generate the draws of the BGCs

```
def SvgDrawGC(self):
    import svgwrite
    import re
    from collections import namedtuple
    from operator import itemgetter
    from itertools import groupby

    DrawType = int(self.favorite.get())
    color = self.favorite2.get()

    x=20
    y=40

    dwg = svgwrite.Drawing(drawname)

    marker = dwg.marker(insert=(0,5), size=(1,1))
    markerini = dwg.marker(insert=(0,5), size=(1,1))
    marker.add(dwg.polyline([(0,0),(0,10),(3,5),(0,0)], stroke=color, fill=color))
    markerini.add(dwg.polyline([(0,0),(0,10),(-3,5),(0,0)], stroke=color, fill=color))
    dwg.defs.add(marker)
    dwg.defs.add(markerini)

    lis_ini = []
    lis_fin = []
    lis_name = []
    try:
        with open(ClusterRange,"r") as f:
            rows = [row.split('\t') for row in f]
            for row in rows:
                lis_name.append(row[0])
                lis_ini.append(row[1])
                lis_fin.append(row[2])
    except NameError:
        ini = self.entry2.get()
        fin = self.entry3.get()
```

```
36            name = self.entry4.get()
37            lis_ini.append(ini)
38            lis_fin.append(fin)
39            lis_name.append(name)
40
41        for ini, fin, name in zip(lis_ini, lis_fin, lis_name):
42            ini = int(ini)
43            fin = int(fin)
44            tam_cluster = (fin-ini)/30.0 ##scale
45            y+=200
46            if DrawType == 0:
47                lll = dwg.add(dwg.line((x,y-20), (x+tam_cluster,y-20), stroke='black',
                      stroke_width=3))
48                texto = dwg.add(dwg.text(name, insert=(x, y-80), fill='black'))
49            elif DrawType == 1:
50                lll = dwg.add(dwg.line((x,y), (x+tam_cluster,y), stroke='black',
                      stroke_width=3))
51                texto = dwg.add(dwg.text(name, insert=(x, y-80), fill='black'))
52            with open(filegbk,"r") as f2:
53                for line in f2:
54                    line = line.strip()
55                    if 'CDS  ' in line:
56                        ran1 = re.findall("(\d+)\.", line)
57                        ran1 = str(ran1)
58                        ran1 = re.sub("\['","",ran1)
59                        ran1 = re.sub("'\]","",ran1)
60                        ran1 = int(ran1)
61                        ran2 = re.findall("\.(\d+)",line)
62                        ran2 = str(ran2)
63                        ran2 = re.sub("\['","",ran2)
64                        ran2 = re.sub("'\]","",ran2)
65                        ran2 = int(ran2)
66                        cds_tam = ((ran2 - ran1)/30.0)-6 ## /30 for scale
67                        d = (ran1-ini)/30.0
68                        d2 = d+cds_tam
69                        plin = next(f2)
70                        sle = re.findall('gene="(.+)"',plin)
71                        if sle == []:
72                            plin2 = next(f2)
73                            sle = re.findall('gene="(.+)"',plin2)
74                        elif sle != []:
75                            pass
76                        sle = str(sle)
77                        sle = re.sub("\['","",sle)
78                        sle = re.sub("'\]","",sle)
79                        if 'complement' not in line:
80                                if ran1 >= ini and ran2 <= fin:
81                                    obj = dwg.add(dwg.rect((x+d,y-27.5),(cds_tam,15),
                                          stroke=color, fill =color,stroke_width=2))
82                                    lin = dwg.add(dwg.line((x+d,y-20),(x+d2,y-20), stroke=
                                          color,stroke_width=2))
83                                    lin['marker-end'] = marker.get_funciri()
84                                    texto = dwg.text(sle, insert=(x+d+cds_tam/2, y-30),
                                          fill='black')
85                                    texto.rotate(-55, center=(x+d+cds_tam/2, y-30))
86                                    dwg.add(texto)
87                        elif 'complement' in line:
88                            if DrawType == 0:
89                                if ran1 >= ini and ran2 <= fin:
90                                    obj = dwg.add(dwg.rect((x+d,y-27.5),(cds_tam,15),
                                          stroke=color, fill =color,stroke_width=2))
91                                    lin = dwg.add(dwg.line((x+d,y-20),(x+d2,y-20), stroke=
                                          color,stroke_width=2))
92                                    lin['marker-start'] = markerini.get_funciri()
93                                    texto = dwg.text(sle, insert=(x+d+cds_tam/2, y-30),
                                          fill='black')
94                                    texto.rotate(-55, center=(x+d+cds_tam/2, y-30))
95                                    dwg.add(texto)
96                            elif DrawType == 1:
97                                if ran1 >= ini and ran2 <= fin:
```

```
                                    obj = dwg.add(dwg.rect((x+d,y+12.5),(cds_tam,15),
                                        stroke=color, fill =color,stroke_width=2))
                                    lin = dwg.add(dwg.line((x+d,y+20),(x+d2,y+20), stroke=
                                        color,stroke_width=2))
                                    lin['marker-start'] = markerini.get_funciri()
                                    texto = dwg.text(sle, insert=(x+d+cds_tam/2-22.8495, y
                                        +75), fill='black')
                                    texto.rotate(-55, center=(x+d+cds_tam/2-22.8495, y+75)
                                        )
                                    dwg.add(texto)
        dwg.save()
```

E.1.2 Draw of the BGCs of *S. leeuwenhoekii*

The following draws were obtained with the code of Section E.1.1. Some of the BGCs were modified with different colors in Inkscape (https://inkscape.org).

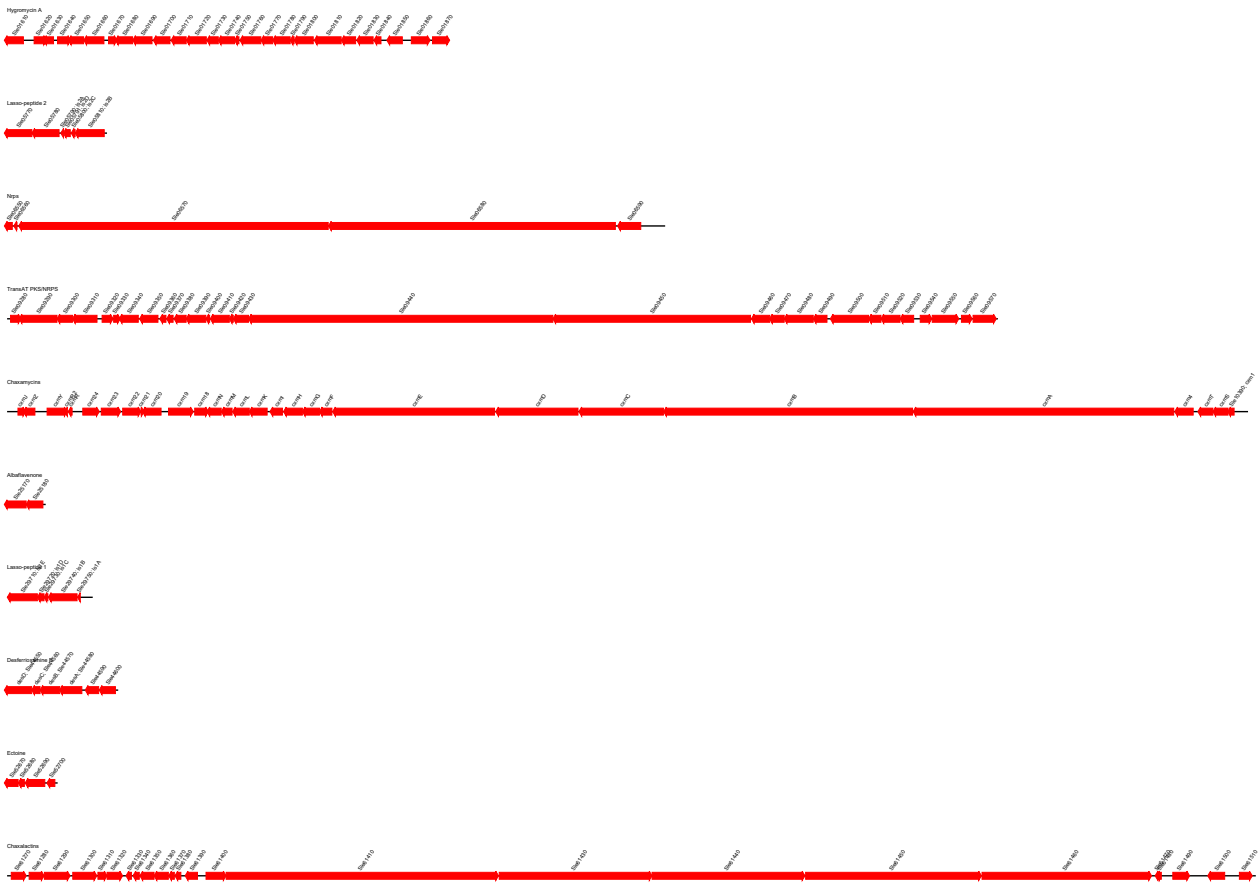


Figure E.1: Draw of the identified BGC of *S. leeuwenhoekii*. The draw was done with the script of Section E.1.1 and, if necessary, edited with Inkscape (https://inkscape.org).

E.2 Alignment of the lasso-peptides cyclization proteins

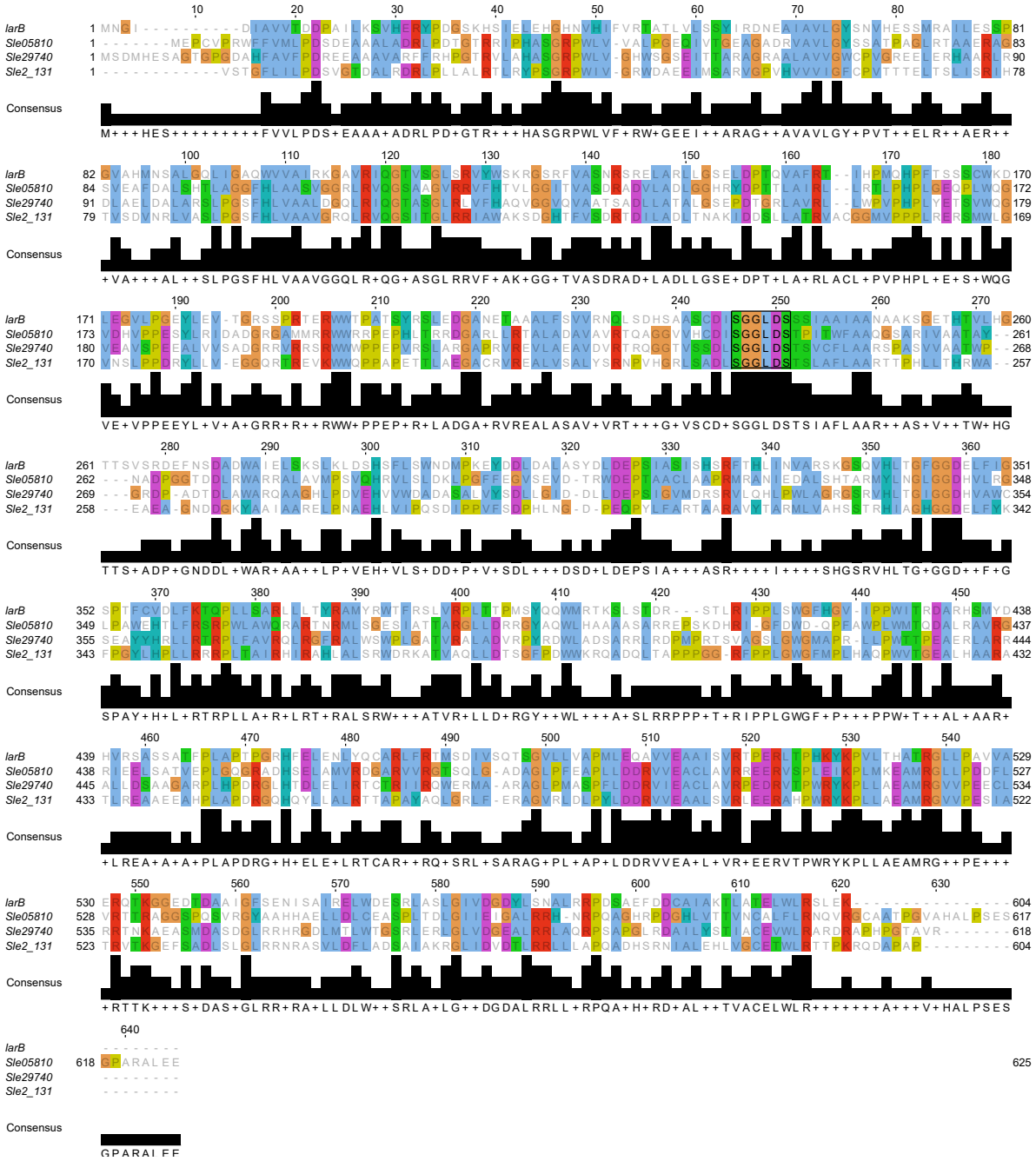


Figure E.2: Alignment of the amino-acids of the proteins Sle29740, Sle05810 and Sle2_131, and their homologous protein of lariatin BGC. The black box indicates the conserved ATP binding pocket motif. LarB (BAL72547)

E.3 Similarities of the hybrid *trans*-AT PKS/NRPS proteins

Table E.1: Description of the proteins found in the hybrid *trans*-AT PKS/NRPS, with closest NCBI database and leinamycin BGC homologous.

Gene	Length (aa)	Proposed function, NRPS/PKS modules, and protein domains	Homologous protein (accession number[1]) (similarity/identity %)	Leinamycin[2] (accession number) (similarity/identity %)
sle09570	500	Major facilitator superfamily (MFS) transporter	MFS transporter *Conexibacter woesei* (WP_012934772) (67/51)	LnmY (AAN85538) (58/41)
sle09560	230	TetR Family Transcripcional Regulator	TetR family transcriptional regulator *Streptomyces tsukubensis* (WP_040914130) (60/47)	
sle09550	572	Long-chain fatty-acid CoA ligase	Acyl-CoA synthetase (AMP-forming)/AMP-acid ligase II, partial *Streptomyces* sp. Termitarium-T10T-6 (SCE60322) (63/50)	LnmW (AAN85536) (56/41)
sle09540	252	Methyltransferase type: MT	methyltransferase *Streptomyces* sp. SPB074 (EFG64226) (64/53)	
sle09530	293	Unknown	hypothetical protein *Streptomyces canus* (WP_063895739) (66/51)	LnmE (AAN85518) (62/48)
sle09520	411	3-hydroxy-3-methylglutaryl-ACP synthase: HMGS	3-hydroxy-3-methylglutaryl-ACP synthase *Saccharothrix* sp. ST-888 (WP_045300246) (78/67)	LnmM (AAN85526) (64/47)
sle09510	268	enoyl-CoA hydratase pksH: ECH	enoyl-CoA hydratase *Streptomyces canus* (WP_059300328) (68/55)	LnmF (AAN85519) (63/50)
sle09500	823	Trans acyltransferase and oxidoreductase: AT1, ER	[acyl-carrier-protein] S-malonyltransferase *Saccharothrix* sp. ST-888 (WP_045300255) (68/58)	LnmG (AAN85520) (78/67)
sle09490	297	Unknown	hypothetical protein *Streptomyces sporocinereus* (WP_062008485) (99/99)	LnmH (AAN85521) (60/44)
sle09480	626	NRPS Module 1: A1, PCP1	amino acid adenylation domain protein *Catenulispora acidiphila* DSM 44928 (ACU71506) (64/54)	Orf-8 (AAN85506) (53/44)
sle09470	316	Chlorinating enzyme	chemotaxis protein CheX *Streptomyces* sp. NRRL S-1022 (WP_030348699) (84/78)	

Continuation of Table E.1

Gene	**Length (aa)**	**Proposed function, NRPS/PKS modules, and protein domains**	**Homologous protein (accession number[1]) (similarity/identity %)**	**Leinamycin[2] (accession number) (similarity/identity %)**
sle09460	397	Alpha/Beta Hydrolase Fold	alpha/beta hydrolase fold protein *Catenulispora acidiphila* DSM 44928 (ACU71508) (64/52)	LnmD () (28)
sle09450	4258	NRPS/PKS PCP2 Module 2: Cy1, Cy2, A2, PCP3, Ox Module 3: KS1^0, ACP1, KS2, KR1, ACP2 Module 4a: KS3	hybrid nonribosomal peptide synthetase / polyketide synthase Streptomyces atroolivaceus (AAN85522) (59/49)	LnmI (AAN85522) (59/49)
sle09440	6558	PKS Module 4b: DH1, ACP3, KR2 Module 5: KS4, ECH, ACP4 Module 6: KS4, KR3, ACP5 Module 7: KS5, KR4, ACP6 Module 8: KS6, ACP7, ACP8, DUF, SH, TE	polyketide synthase *Saccharothrix espanaensis* (WP_015102008) (58/48)	LnmJ (AAN85523) (61/50)
sle09430	327	Bifunctional Acyltransferase/Decarboxylase	biosynthesis cluster domain-containing protein *Streptomyces* sp. Termitarium-T10T-6 (SCE60091) (66/56)	LnmK (4HZO_A) (64/52)
sle09420	86	Discrete ACP: ACP9	polyketide biosynthesis acyl carrier protein *Streptomyces* sp. Termitarium-T10T-6 (SCE60094) (75/59)	LnmL (AAN85525) (67/50)
sle09410	410	3-hydroxy-3-methylglutaryl-ACP synthase: HMGS	3-hydroxy-3-methylglutaryl-ACP synthase *Streptomyces specialis* (WP_059006297) (79/69)	LnmM (AAN85526) (76/64)
sle09400	82	Discrete ACP: ACP10	phosphopantetheine-binding protein *Streptomyces hygroscopicus* (WP_060949990) (97/95)	LnmL (AAN85525) (62/37)
sle09390	424	Beta-ketoacyl-acyl synthase	beta-ketoacyl synthase *Salinispora arenicola* (WP_018792602) (65/55)	

Continuation of Table E.1

Gene	Length (aa)	Proposed function, NRPS/PKS modules, and protein domains	Homologous protein (accession number[1]) (similarity/identity %)	Leinamycin[2] (accession number) (similarity/identity %)
sle09380	260	Thioesterase: TE	thioesterase *Streptomyces canus* (WP_059300315) (67/53)	LnmN (AAN85527) (64/52)
sle09370	139	Unknown	hypothetical protein *Streptacidiphilus rugosus* (WP_037601152) (68/45)	LnmV (AAN85535) (61/43)
sle09360	133	Unknown	hypothetical protein *Streptomyces* sp. NBRC 109436 (WP_064455879) (98/98)	LnmZ' (AAN85540) (48/34)
sle09350	396	Cytochrome P450-SU2	cytochrome P450 *Catenulispora acidiphila* (WP_012786793) (72/59)	LnmA (4Z5P_A) (57/43)
sle09340	411	Cytochrome P450 107B1	cytochrome P450 *Allokutzneria albata* (WP_030426450) (68/53)	LnmA (4Z5P_A) (66/50)
sle09330	144	Glyoxalase/Bleomycin Resistance Protein/Dioxygenase	glyoxalase *Mycobacterium* sp. 852002-51057_SCH5723018 (OBG27001) (51/42)	
sle09320	238	N-acetylglucosaminyl deacetylase	GlcNAc-PI de-N-acetylase *Catenulispora acidiphila* (WP_012786815) (77/67)	LnmX (5BMO_A) (65/53)
sle09310	530	ABC-type transport system, periplasmic component	Oligopeptide-binding protein AppA *Streptomyces afghaniensis* (WP_020270582) (64/46)	LnmU (AAN85534) (46/32)
sle09300	333	ABC-type dipeptide/oligopeptide/nickel transport system, permease component	binding-protein-dependent transport systems inner membrane component *Streptomyces lincolnensis* (ANS63568) (72/58)	LnmT (AAN85533) (61/44)
sle09290	817	ABC-type Glutathion transport system: ATPase component ABC-type dipeptide/oligopeptide/nickel transport system: permease component	ATPase component of various ABC-type transport systems with duplicated ATPase domain protein *Streptomyces lincolnensis* (ANS63567) (62/51)	LnmR (AAN85531) (55/42)
sle09280	226	Transcriptional activator	Crp/Fnr family transcriptional regulator *Catenulispora acidiphila* (WP_012786802) (65/47)	LnmO (AAN85528) (68/49)

[1]NCBI accession number.

[2]Leinamycin BGC found in *S. atroolivaceus* (NCBI accession number: AF484556).

E.4 Alignment of the hybrid *trans*-AT PKS/NRPS domains and regulator genes

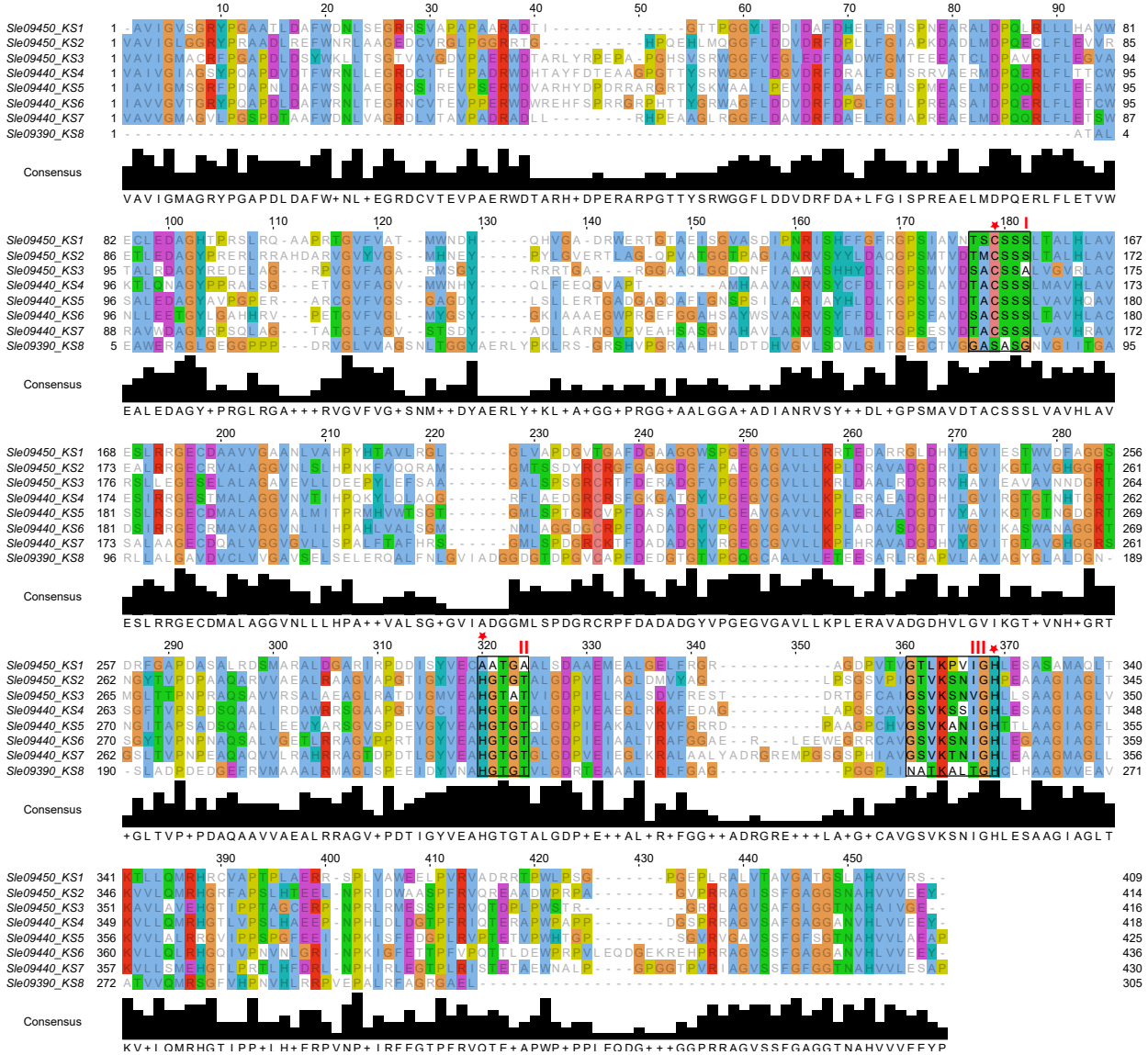


Figure E.3: Alignment of the KS domains of the hybrid *trans*-AT PKS/NRPS BGC. The conserve motifs TxCxxS, HGTGT and GSVKxxxGH, are shown in box I, II and III, respectively. The catalytic residues are highlighted with red stars.

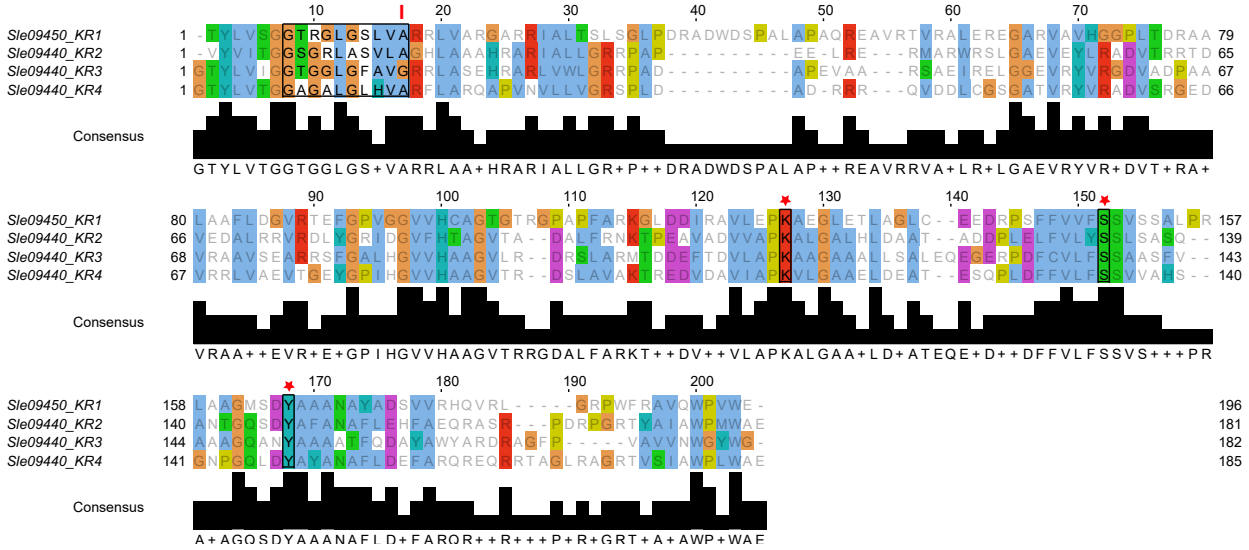


Figure E.4: Alignment of the KR domains of the hybrid *trans*-AT PKS/NRPS BGC. The conserved motif for NADP(H) binding, GxGxxGxxxA, is shown in box I. The conserved catalytic triad, lysine-serine-tyrosine, necessary for ketoreduction is highlighted with red stars.

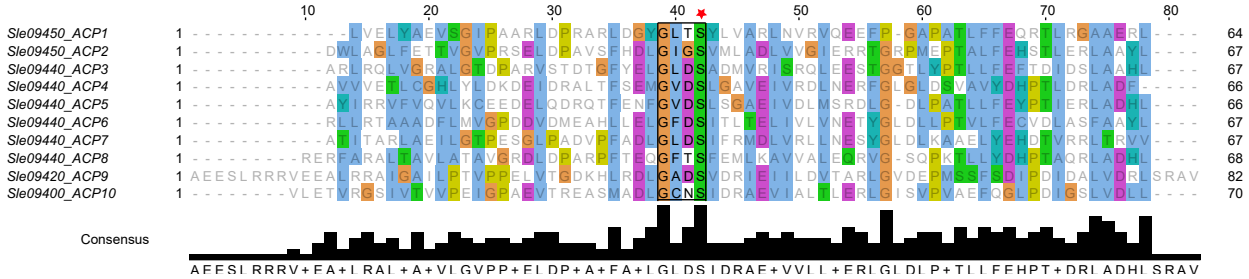


Figure E.5: Alignment of the ACP domains of the hybrid *trans*-AT PKS/NRPS BGC. The conserved serine, necessary for phosphopantetheine attachment is highlighted with a red star.

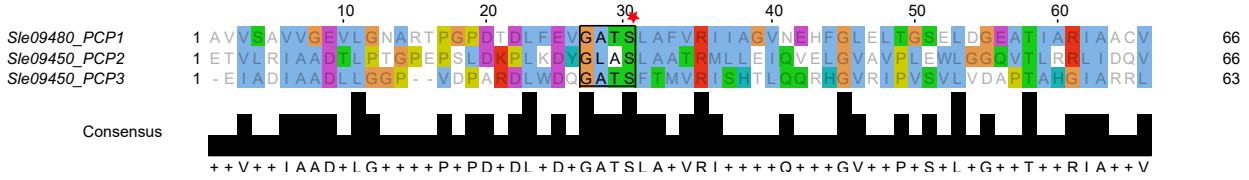


Figure E.6: Alignment of the PCP domains of the hybrid *trans*-AT PKS/NRPS BGC. The conserved serine, necessary for phosphopantetheine attachment is highlighted with a red star.

E.5 Supplementary chromatograms

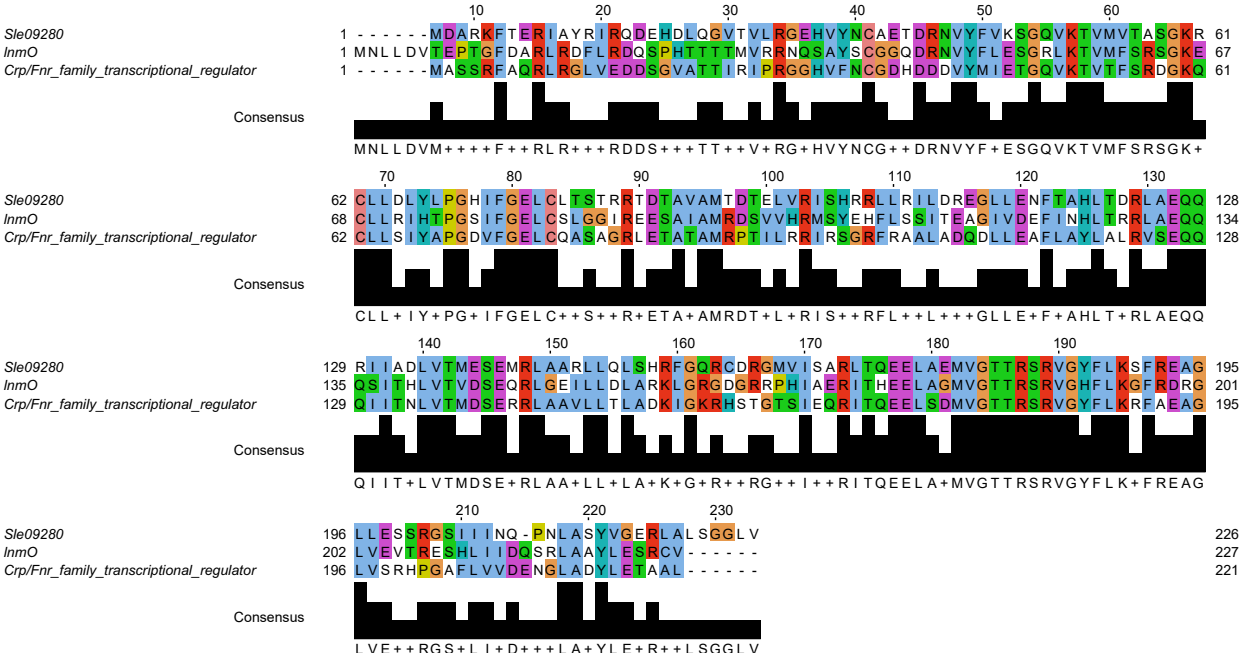


Figure E.7: Alignment of the amino-acids of the putative regulator enzyme (Sle09280) detected in the hybrid *trans*-AT PKS/NRPS BGC, with similar known proteins: lnmO from *S. atroolivaceus* (NCBI accession number: AAN85528) and Crp/Fnr transcriptional regulator from *Catenulispora acidiphila* (NCBI accession number: WP_012786802).

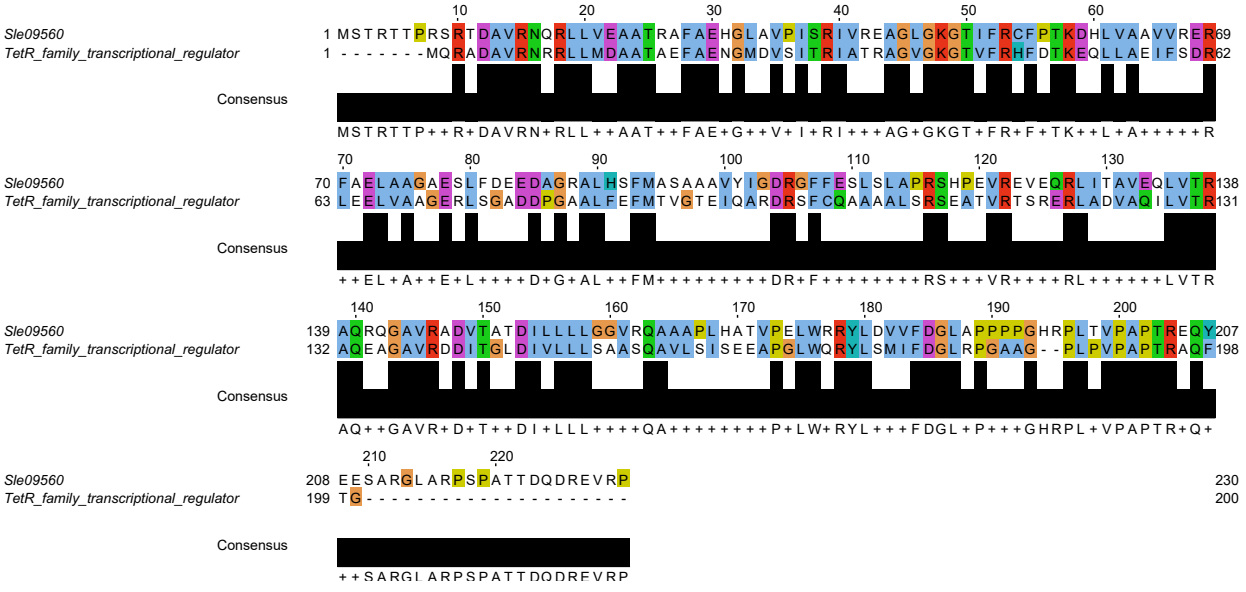


Figure E.8: Alignment of the amino-acids of the putative regulator enzyme (Sle09560) detected in the hybrid *trans*-AT PKS/NRPS BGC, with TetR transcriptional regulator protein from *S. tsukubensis* (NCBI accession number: WP_040914130)

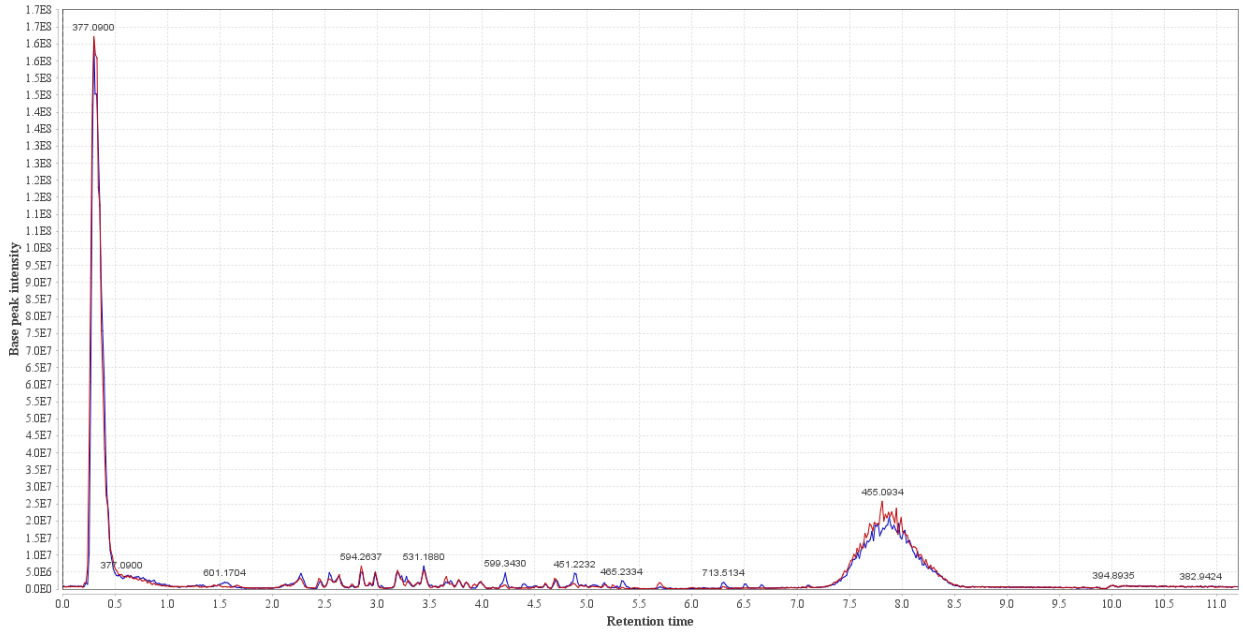


Figure E.9: Metabolite profile comparison between *S. coelicolor* M1152 (blue) and *S. coelicolor* M1607 (red).

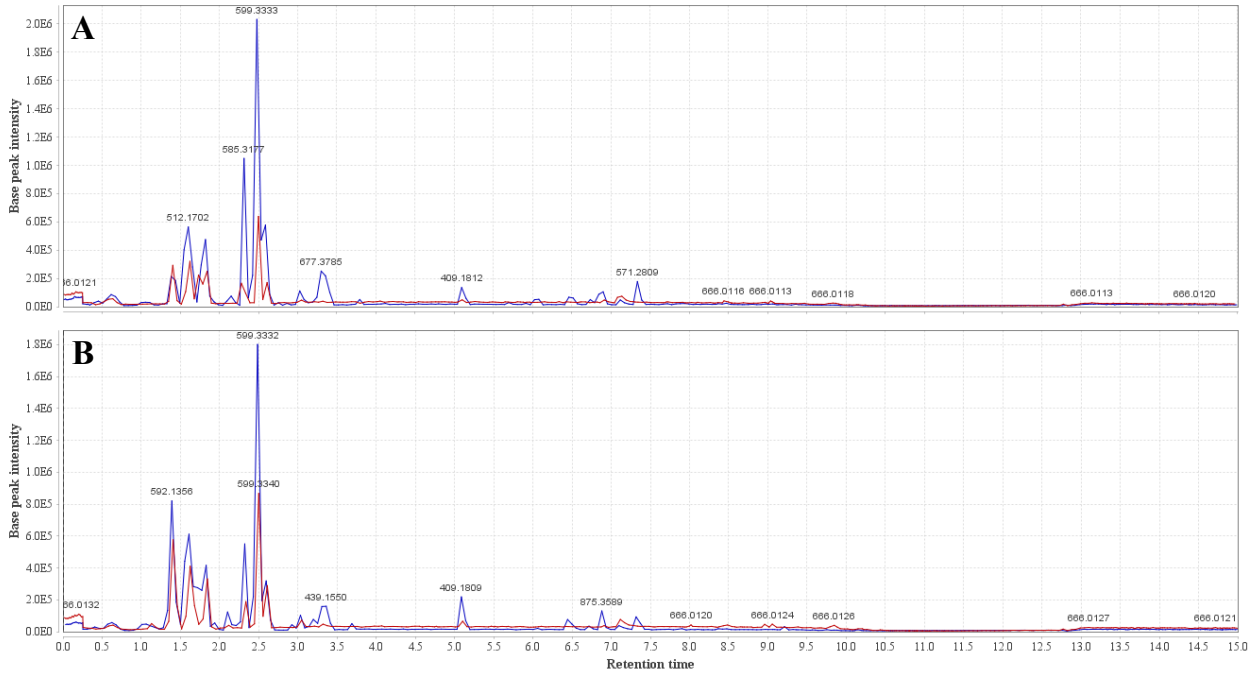


Figure E.10: Chromatograms of samples extracted with acetone-methanol (blue) and with methanol (red) from *S. leeuwenhoekii* M1614 (**A**) and *S. leeuwenhoekii* M1619 **B**.

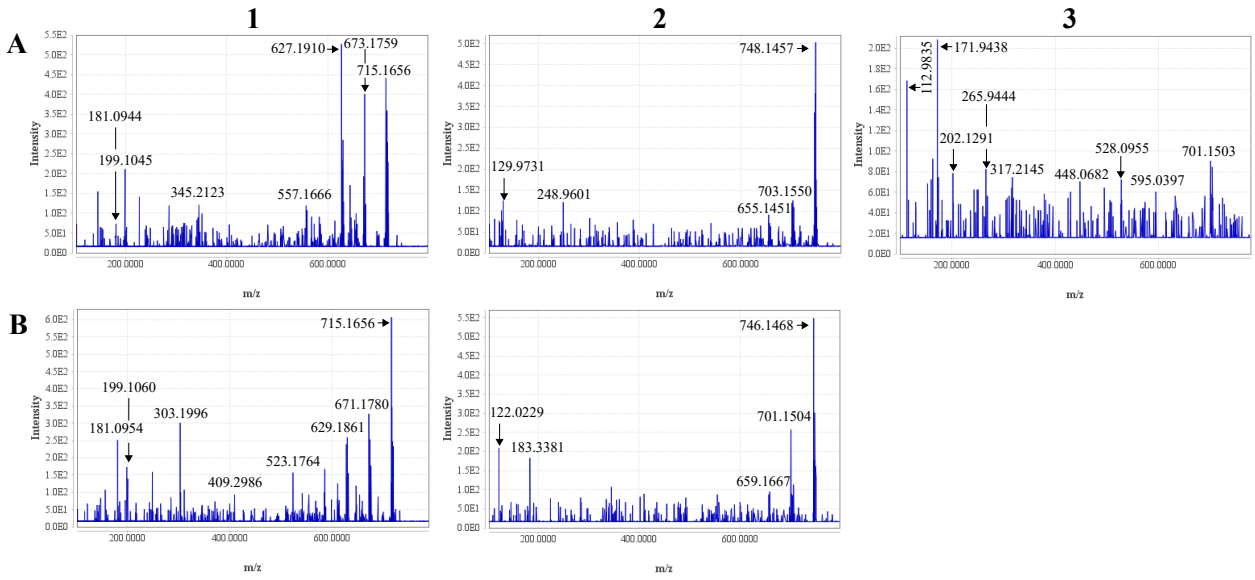


Figure E.11: Fragmentation pattern of the ions m/z 715.16 (**1**), 746.14 (**2**) and 791.14 (**3**) observed in the samples of supernatant and mycelium extract of *S. leeuwenhoekii* M1614 (**A**) and M1619 (**B**). The samples were run with negative ionization in a Maxis II qTOF LCMS. Mass spectrum was visualized with Mzmine (Pluskal et al., 2010).

Appendix F

Supplementary information for Chapter Two

F.1 Formulation of the biomass equation and their components

The biomass equation of *S. leeuwenhoekii* was based on (Borodina et al., 2005). The equations for production of biomass building blocks were based on information of *S. leeuwenhoekii* when possible, and complemented with information of either *S. coelicolor* (Borodina et al., 2005) or *S. tsukubaensis* (Huang et al., 2013), *E. coli* (Orth et al., 2011) or other *Streptomyces*.

F.1.1 Biomass

The biomass equation at 0.109 h^{1} dilution rate is given by:

0.412 Protein + 0.167 RNA + 0.036 DNA + 0.027 Phospholipid + 0.018 TAG + 0.03 Small molecules + 0.110 Peptidoglycan + 0.044 Carbohydrate + 0.066 Teichoich acid + 47 ATP $\rightarrow$ Biomass + 47 ADP + 47 Orthophosphate + 47 H^{+}

F.1.2 Protein

The amino acid composition of *S. leeuwenhoekii* was not available in the literature. Thus, it was assumed that protein composition of *S. leeuwenhoekii* is the same as *S. tsukubaensis* (Huang et al., 2013). Energy requirement for polymerisation was taken as for *E. coli* (Ingraham et al., 1983).

Accordingly, the protein equation is:

Table F.1: Components of the biomass equation at a dilution rate of 0.109 h ^{1}

Component	$\mathbf{g \cdot g^{-1}}$ **DW**
Protoplast	0.690
Protein	0.412
DNA	0.036
RNA	0.167
Lipids	0.045
Polar (Phospholipids)	0.027
Non-polar (TAGs)	0.018
Small molecules	0.030
Cell wall	0.220
Peptidoglycan	0.110
Carbohydrates	0.044
Teichoich acid	0.066
Ash	0.090
SUM	1.000

Table F.2: Aminoacid composition used for *S. leeuwenhoekii* based on aminoacid composition of *S. tsukubaensis* (Huang et al., 2013). considering the energy required for polymerization of *E. coli* (Ingraham et al., 1983)

Amino acid	Abbreviation	% protein (w/w)	MW^{1} (g/mol)	mmol/g protein
Alanine	LAla_c	10.52	71.094	1.480
Arginine	LArg_c	3.89	156.203	0.249
Asparagine	LAspa_c	5.08	114.119	0.445
Aspartate	LAsp_c	5.15	115.104	0.447
Cysteine	LCys_c	2.02	103.160	0.196
Glutamate	LGlu_c	7.80	128.146	0.609
Glutamine	LGlut_c	4.41	129.131	0.342
Glycine	Gly_c	7.66	57.067	1.342
Histidine	LHis_c	2.66	137.156	0.194
Isoleucine	LIsoleu_c	6.82	113.175	0.603
Leucine	LLeu_c	3.84	113.175	0.339
Lysine	LLys_c	2.22	128.189	0.173
Methionine	LMet_c	3.84	131.214	0.293
Phenylalanine	LPhenylala_c	4.83	147.192	0.328
Proline	LPro_c	8.80	97.132	0.906
Serine	LSer_c	4.51	87.093	0.518
Threonine	LThr_c	6.55	101.120	0.648
Tryptophan	LTry_c	1.48	186.228	0.079
Tyrosine	LTyr_c	2.59	163.191	0.159
Valine	LVal_c	5.33	99.148	0.538
Energy requirement for polymerisation (ATP):				40

1.480 LAla_c + 0.249 LArg_c + 0.445 LAspa_c + 0.447 LAsp_c + 0.196 LCys_c + 0.609 LGlu_c + 0.342 LGlut_c + 1.342 Gly_c + 0.194 LHis_c + 0.603 LIsoleu_c + 0.339 LLeu_c + 0.173 LLys_c + 0.293 LMet_c + 0.328 LPhenylala_c + 0.906 LPro_c + 0.518 LSer_c + 0.648 LThr_c + 0.079 LTry_c + 0.159 LTyr_c + 0.538 LVal_c + 40.0 ATP_c $\rightarrow$ Protein_c + 40.0 ADP_c + 40.0 pi_c

F.1.3 RNA

RNA composition was assumed to be 5 % mRNA, 75 % rRNA and 20 % tRNA. The nucleotide composition of mRNA was taken as for genomic DNA (see F.1.4). The nucleotide composition of rRNA was calculated from the sequences of 16S, 23S and 5S ribosomal RNA units. tRNA composition was found from sequences of transporting RNAs. The energy requirement for polymerisation of triphosphates was taken from (Ingraham et al., 1983).

Table F.3: RNA composition of *S. leeuwenhoekii*.

Nucleotide	mol/mol	MW[1] (g/mol)	mmol/g RNA
AMP	0.211	329.2	0.6509
GMP	0.340	345.2	1.0484
CMP	0.254	305.2	0.785
UMP	0.195	306.2	0.6017
Energy requirement for polymerisation (ATP):			1.25

Therefore, the RNA equation is:

0.6509 ATP + 1.0484 GTP + 0.785 CTP + 0.6017 UTP + 1.25 ATP $\rightarrow$ 1.25 ADP + 1.25 Orthophosphate + RNA + 3.086 Diphosphate

F.1.4 DNA

DNA composition was calculated from the genome sequence of *S. leeuwenhoekii* (NCBI accesion number: http://www.ncbi.nlm.nih.gov/bioproject/284471). The energy requirement for polymerisation of triphosphates was taken from (Ingraham et al., 1983).

Table F.4: DNA composition of *S. leeuwenhoekii*.

Nucleotide	mol/mol	MW[1] (g/mol)	mmol/g DNA
dAMP	0.136	313.2	0.440
dCMP	0.363	289.2	1.175
dTMP	0.136	304.2	0.441
dGMP	0.364	329.2	1.179
Energy requirement for polymerisation (ATP):			4.40

Hence the DNA equation is:

0.4404 dATP + 1.1791 dGTP + 1.1751 dCTP + 0.4407 dTTP + 4.40 ATP $\rightarrow$ 4.40 ADP + 4.40 Orthophosphate + DNA + 3.2353 Diphosphate

F.1.5 Phospholipid and TAG composition

Table F.5: Fatty acid composition of *S. leeuwenhoekii* (Busarakam, 2014) and proportion present in phospholipids and in TAGs.

Fatty acid	% mol/mol	Cardiolipin		Phosphatidylethanolamine		Phosphatidylinositol		Phosphatidic acid		Triacylglycerol	
		Abbreviation	mmol/g	Abbreviation	mmol/g	Abbreviation	mmol/g	Abbreviation	mmol/g	Abbreviation	mmol/g
Isotridecanaoate (iso-C13:0)	0.4	clpniC130_c	9.011E-04	pei130_c	3.225E-03	pii130_c	2.455E-04	pai130_c	6.139E-04	tagi130_c	4.956E-03
Anteisotridecanaoate (anteiso C13:0)	0.2	clpnaiC130_c	4.505E-04	peai130_c	1.613E-03	piai130_c	1.228E-04	paai130_c	3.069E-04	tagai130_c	2.478E-03
Isotetradecanaoate (iso-C14:0)	4.3	clpni140_c	9.686E-03	pei140_c	3.467E-02	pii140_c	2.639E-03	pai140_c	6.599E-03	tagi140_c	5.328E-02
Isopentadecanaoate (iso-C15:0)	5.5	clpni150_c	1.239E-02	pei150_c	4.434E-02	pii150_c	3.376E-03	pai150_c	8.441E-03	tagi150_c	6.815E-02
Anteisopentadecanaoate (anteiso-C15:0)	29.2	clpnai150_c	6.578E-02	peai150_c	2.354E-01	piai150_c	1.792E-02	paai150_c	4.481E-02	tagai150_c	3.618E-01
isoC16:0/isoC16:1H	12.9	clpni160_c	2.906E-02	pei160_c	1.040E-01	pii160_c	7.918E-03	pai160_c	1.980E-02	tagi160_c	1.598E-01
Anteisoheptadecenaoate (anteiso-C17:1w9c)	1.1	clpnaiC171_c	2.478E-03	peai171_c	8.869E-03	piai171_c	6.751E-04	paai171_c	1.688E-03	tagai171_c	1.363E-02
isoC17:0	4.4	clpni170_c	9.912E-03	pei170_c	3.548E-02	pii170_c	2.701E-03	pai170_c	6.753E-03	tagi170_c	5.452E-02
anteisoC17:0	13.8	clpnai170_c	3.109E-02	peai170_c	1.113E-01	piai170_c	8.470E-03	paai170_c	2.118E-02	tagai170_c	1.710E-01
isoC17:1w9c/10methylC16:0	1.0	clpniC171_c	2.253E-03	pei171_c	8.063E-03	pii171_c	6.138E-04	pai171_c	1.535E-03	tagi171_c	1.239E-02
isoC18:0	0.5	clpni180_c	1.126E-03	pei180_c	4.031E-03	pii180_c	3.069E-04	pai180_c	7.673E-04	tagi180_c	6.195E-03
C12:0	0.2	clpnC120_c	4.505E-04	pe120_c	1.613E-03	pi120_c	1.228E-04	pa120_c	3.069E-04	tag120_c	2.478E-03
C14:0	1.1	clpnC140_c	2.478E-03	pe140_c	8.869E-03	pi140_c	6.751E-04	pa140_c	1.688E-03	tag140_c	1.363E-02
C15:0 2OH*	0.2	clpn150_c	4.505E-04	pe150_c	1.613E-03	pi150_c	1.228E-04	pa150_c	3.069E-04	tag150_c	2.478E-03
C16:0	19.1	clpnC160_c	4.303E-02	pe160_c	1.540E-01	pi160_c	1.172E-02	pa160_c	2.931E-02	tag160_c	2.367E-01
C16:1w7c/C16:1w6c	1.5	clpn161_c	3.379E-03	pe161_c	1.209E-02	pi161_c	9.206E-04	pa161_c	2.302E-03	tag161_c	1.859E-02
C17:0	3.5	clpn170_c	7.884E-03	pe170_c	2.822E-02	pi170_c	2.148E-03	pa170_c	5.371E-03	tag170_c	4.337E-02
C17:1w8c	0.2	clpnC171_c	4.505E-04	pe171_c	1.613E-03	pi171_c	1.228E-04	pa171_c	3.069E-04	tag171_c	2.478E-03
C18:0	0.9	clpnC180_c	2.027E-03	pe180_c	7.256E-03	pi180_c	5.524E-04	pa180_c	1.381E-03	tag180_c	1.115E-02

9.011E-04 clpniC130_c + 4.505E-04 clpnaiC130_c + 9.686E-03 clpni140_c + 1.239E-02 clpni150_c + 6.578E-02 clpnai150_c + 2.906E-02 clpni160_c + 2.478E-03 clpnaiC171_c + 9.912E-03 clpni170_c + 3.109E-02 clpnai170_c + 2.253E-03 clpniC171_c + 1.126E-03 clpni180_c + 4.505E-04 clpnC120_c + 2.478E-03 clpnC140_c + 4.505E-04 clpn150_c + 4.303E-02 clpnC160_c + 3.379E-03 clpn161_c + 7.884E-03 clpn170_c + 4.505E-04 clpnC171_c + 2.027E-03 clpnC180_c + 3.225E-03 pei130_c + 1.613E-03 peai130_c + 3.467E-02 pei140_c + 4.434E-02 pei150_c + 2.354E-01 peai150_c + 1.040E-01 pei160_c + 8.869E-03 peai171_c + 3.548E-02 pei170_c + 1.113E-01 peai170_c + 8.063E-03 pei171_c + 4.031E-03 pei180_c + 1.613E-03 pe120_c + 8.869E-03 pe140_c + 1.613E-03 pe150_c + 1.540E-01 pe160_c + 1.209E-02 pe161_c + 2.822E-02 pe170_c + 1.613E-03 pe171_c + 7.256E-03 pe180_c + 2.455E-04 pii130_c + 1.228E-04 piai130_c + 2.639E-03 pii140_c + 3.376E-03 pii150_c + 1.792E-02 piai150_c + 7.918E-03 pii160_c + 6.751E-04 piai171_c + 2.701E-03 pii170_c + 8.470E-03 piai170_c + 6.138E-04 pii171_c + 3.069E-04 pii180_c + 1.228E-04 pi120_c + 6.751E-04 pi140_c + 1.228E-04 pi150_c + 1.172E-02 pi160_c + 9.206E-04 pi161_c + 2.148E-03 pi170_c + 1.228E-04 pi171_c + 5.524E-04 pi180_c + 6.139E-04 pai130_c + 3.069E-04 paai130_c + 6.599E-03 pai140_c + 8.441E-03 pai150_c + 4.481E-02 paai150_c + 1.980E-02 pai160_c + 1.688E-03 paai171_c + 6.753E-03 pai170_c + 2.118E-02 paai170_c + 1.535E-03 pai171_c + 7.673E-04 pai180_c + 3.069E-04 pa120_c + 1.688E-03 pa140_c + 3.069E-04 pa150_c + 2.931E-02 pa160_c + 2.302E-03 pa161_c + 5.371E-03 pa170_c + 3.069E-04 pa171_c + 1.381E-03 pa180_c → 1 Plipid_c

4.956E-03 tagi130_c + 2.478E-03 tagai130_c + 5.328E-02 tagi140_c + 6.815E-02 tagi150_c + 3.618E-01 tagai150_c + 1.598E-01 tagi160_c + 1.363E-02 tagai171_c + 5.452E-02 tagi170_c + 1.710E-01 tagai170_c + 1.239E-02 tagi171_c + 6.195E-03 tagi180_c + 2.478E-03 tag120_c + 1.363E-02 tag140_c + 2.478E-03 tag150_c + 2.367E-01 tag160_c + 1.859E-02 tag161_c + 4.337E-02 tag170_c + 2.478E-03 tag171_c + 1.115E-02 tag180_c → 1 TAG_c

F.1.6 Small molecules pool

Table F.6: Small molecules pool.

Molecule	Abbreviation	mmol/g
NAD	NAD_c	0.188
NADP	NADP_c	0.167
CoA	CoA_c	0.162
Acyl-carrier protein	ACP_c	0.011
Menaquinone	Men_c	0.146
Tetrahydrofolate	Thy_c	0.280
FMN	FMN_c	0.273
FAD	FAD_c	0.159

Therefore, the equation for synthesis of small molecules is:

0.188 NAD_c + 0.168 NADP_c + 0.163 CoA_c + 0.012 ACP_c + 0.146 Men_c + 0.281 Thy_c + 0.274 FMN_c + 0.159 FAD_c $\rightarrow$ 1 Smallmol_c

F.1.7 Peptidoglycan synthesis

Peptidoglycan composition was calculated as the average of several *Streptomyces* strains (Streshinskaia et al., 1978; Skoblilova et al., 1982; Zaretskaia and Polin, 1987), like it was done in the model of *S. coelicolor* of (Borodina et al., 2005).

Table F.7: Peptidoglycan composition.

Component	Abbreviation	Molar ratio[1]	MW[2] (g/mol)	mmol/g
N-acetylmuramic acid	UDPNace_c	0.883	275.3	1.007
N-acetylglucosamine	UDPNaceaDglu_c	1.050	203.2	1.197
Alanine	DAlanylDala_c	1.667	71.1	1.900
Diaminopimelinic acid	mes26Diate_c	1.000	154.2	1.140
D-glutamate	Dgluta_c	0.890	129.1	1.014
Glycine	Gly_c	0.853	39.0	0.973
Energy requirement for polymerisation (ATP):				5.026

[1]The molar ratio showed correspond to the average of several *Streptomyces*.
[2]The molecular weight is subtracted water to account for the bond formation, glycine and Diaminopimelinic acid are subtracted two molecules of H_2O each to account for cross-linking as well.

Accordingly, the peptidoglycan composition equation is:

1.197 UDPNaceaDglu_c + 1.007 UDPNace_c + 1.900 DAlanylDala_c + 1.140 mes26Diate_c + 1.014 DGluta_c + 0.973 Gly_c + 5.026 ATP_c $\rightarrow$ 1 Peptidoglycan_c + 0.950 DAla_c + 1.197 UDP_c + 1.007 UMP_c + 5.026 ADP_c + 5.026 pi_c

F.1.8 Carbohydrate biosynthesis

The cell wall carbohydrates composition was assumed to be identical to *S. antibioticus* (Zaretskaia and Polin, 1987). The polysaccharides are made from activated building blocks, therefore there is no need for additional ATP during polymerisation.

Table F.8: Carbohydrates composition of cell wall.

Component	Abbreviation	Molar ratio	MW[1] (g/mol)	mmol/g
N-acetylglucosamine	UDPNaceaDglu_c	1	203.19	1.897
Galactose	UDPaDgal_c	2	162	3.794

[1]The molecular weight is subtracted water to account for the bond formation

The equation for carbohydrate biosynthesis is:

1.897 UDPNaceaDglu_c + 3.794 UDPaDgal_c $\rightarrow$ 1 Carbohyd_c + 5.691 UDP_c

F.1.9 Teichoic acid biosynthesis

The teichoic acid composition was assumed as the same used for *S. coelicolor* (Borodina et al., 2005), where it was assumed that the teichoic acid has the same composition as *S. roseoflavus* var. roseofungini (Skoblilova et al., 1982).

Table F.9: Teichoic acid biosynthesis composition.

Components	Abbreviation	Molar ratio	MW[1] (g/mol)	mmol/g
Polyglycerophosphate chain	pgly_c	1	1848.708	0.518
Lysine	LLys_c	0.25	128.174	0.129
N-acetylglucosamine	UDPNaceaDglu_c	0.25	203.194	0.129
Energy requirement for polymerisation (ATP):				0.129

[1]The molecular weight of polyglycerophosphate chain is calculated as $(C_3H_7O_5P)_{12}$, the molecular weight of lysine and N-acetylglucosamine is subtracted water to account for the bond formation.

Therefore, the equation for teichoic acid composition is:

0.518 pgly_c + 0.129 LLys_c + 0.129 UDPNaceaDglu_c + 0.129 ATP_c $\rightarrow$ 1 Teicad_c + 0.129 UDP_c + 0.129 ADP_c + 0.129 pi_c

F.1.10 Ions pool

The ion pool composition was extracted from the composition described for *E. coli* in (Orth et al., 2011).

Table F.10: Ions pool composition.

Element	Abbreviation	mmol/gDW
Calcium	Ca2_c	0,004952
Chloride	Cl_c	0,004952
Cobalt	Cobalt2_c	0,000024
Copper	Cu2_c	0,000674
Ferrous (2^+)	Fe2_c	0,006388
Ferrous (3^+)	Fe3_c	0,007428
Potassium	K_c	0,185690
Magnesium	Mg2_c	0,008253
Manganese	Mn2_c	0,000658
Molybdate	Molbd_c	0,000007
Ammonium	NH3_c	0,012379
Nickel	Ni_c	0,000307
Sulfate	Sulf_c	0,004126
Zinc	Zn2_c	0,000324

Therefore, the equation for the ion pool is:

0.004952 Ca2_c + 0.004952 Cl_c + 0.000024 Cobalt2_c + 0.000674 Cu2_c + 0.006388 Fe2_c + 0.007428 Fe3_c + 0.185690 K_c + 0.008253 Mg2_c + 0.000658 Mn2_c + 0.000007 Molbd_c + 0.012379 NH3_c + 0.000307 Ni_c + 0.004126 Sulf_c + 0.000324 Zn2_c $\rightarrow$ ions_c

F.1.11 Essential gene list

Table F.11: Essential genes for *S. leeuwenhoekii* C34 for growth in complex media.

Gene
Sle56320
Sle46560
Sle47660
Sle19300
Sle29230
Sle60800
Sle56300
Sle19710
Sle53190
Sle20960
Sle30740
Sle27020
Sle18260
Sle41580
Sle13070
Sle56310
Sle56280
Sle29830
Sle32860
Sle50520

Continuation of Table F.11

Gene
Sle50550
Sle50530
Sle48110
Sle57210
Sle53580
Sle38760
Sle47120
Sle57780
Sle13080
Sle48100
Sle50570
Sle29360
Sle41380
Sle47130
Sle29840
Sle57220
Sle29960
Sle47140
Sle55840
Sle50540
Sle38770
Sle57230
Sle51770
Sle56220
Sle16170
Sle28890
Sle45140
Sle35950
Sle18780
Sle26030
Sle52250
Sle19260
Sle49810
Sle29290
Sle33510
Sle32590
Sle53500
Sle36080
Sle29430
Sle33820
Sle66460
Sle27600
Sle41020
Sle18340
Sle38050
Sle50770
Sle14550
Sle46720

Continuation of Table F.11

Gene
Sle32580
Sle37360
Sle19870
Sle56230
Sle48130
Sle33670
Sle46270
Sle58140

F.1.12 Growth in complex media

Table F.12: Uptake rates of compound used to simulate growth in complex media (ISP2).

Compound	Uptake rate (mmol $gDW^{-1}h^{-1}$)
D-Glucose	-10
Glycerol	-10
Maltose	-10
D-Fructose	-10
Riboflavin	-10
Nicotinate	-10
Pantothenate	-10
Folate	-10
Inositol	-10
Biotin	-10
4-Aminobenzoate	-10
Alpha,alpha-Trehalose	-1
L-Alanine	-0.1
L-Arginine	-0.1
L-Aspartate	-0.1
L-Cysteine	-0.1
L-Glutamate	-0.1
Glycine	-0.1
L-Histidine	-0.1
L-Isoleucine	-0.1
L-Leucine	-0.1
L-Lysine	-0.1
L-Methionine	-0.1
L-Phenylalanine	-0.1
L-Proline	-0.1
L-Serine	-0.1
L-Threonine	-0.1
L-Tryptophan	-0.1
L-Valine	-0.1

F.2 *S. leeuwenhoekii* map of reactions

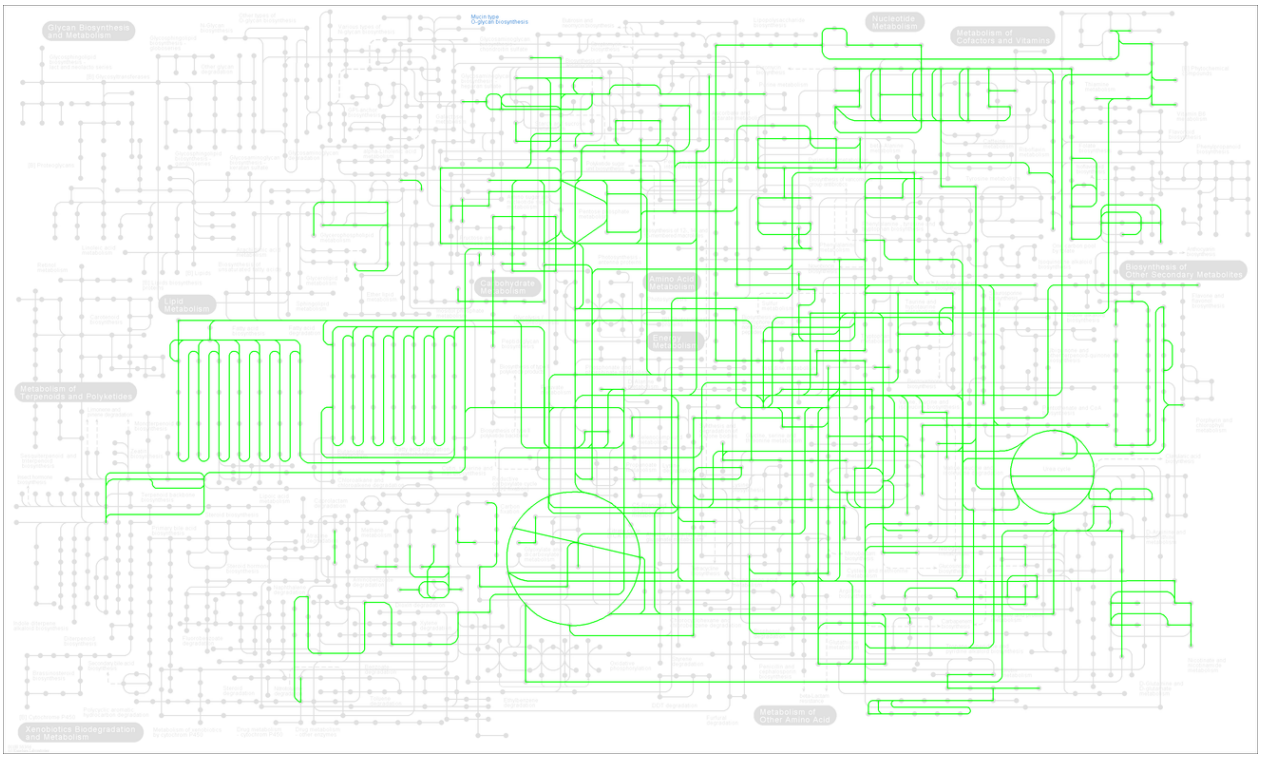


Figure F.1: Map of reactions included in the genome scale model, *i*VR1007, of *Streptomyces leeuwenhoekii*.

F.3 Script used to generate and work with the GSM

F.3.1 Get the reaction information from KEGG database

```
def retrieve(self):
    from bioservices import Kegg
    from bioservices import *
    s = Kegg()
    w = KeggParser()
    import re
    content = self.textinput.get(0.0, END)
    content = str(content)
    gaa_list = []
    info_list = []
    strep_names = ['SFI:','SALB:','SBH:','SCB:','SCI:','SCT:','SCY:','SDV:','SFA:','
        SCO:','SGR:','SHO:','SHY:','SLV:',
                   'SMA:','SRC:','SSX:','STRP:','SVE:','SVL:','SALU:','SGU:','SVT:','
                       STRE:','SCW:']
    gene_search = ['SFUL\_\d{4}','XNR\_\d{4}','SBI\_\d{5}','SCAB\_\d{5}','B446\_\d{5}'
        ,'SCAT\_\d{4}','SCATT\_\d{5}',
                   'BN159\_\d{4}','Sfla\_\d{4}','SCO\d{4}','SGR\_\d{4}','SHJGH\_\d{4}'
                       ,'SHJG\_\d{4}','SLIV\_\d{5}',
                   'SAV\_\d{4}','M271\_\d{5}','SACTE\_\d{4}','F750\_\d{4}','SVEN\_\d
                       {4}','Strvi\_\d{4}','DC74\_\d{4}',
                   'SGLAU\_\d{5}','SVTN\_\d{5}','GZL\_\d{5}','TU94\_\d{5}']
    try:
        self.text.delete(0.0, END)
        self.text2.delete(0.0, END)
        content = re.sub("R","r",content)
        reaction_list = re.findall("(r\d{5})",content)
```

```
            for item in reaction_list:
                ri = s.get('rn:'+item)
                d = w.parseReaction(ri)
                rxn_words = d.get("definition")
                rxn_comp = d.get("equation")
                defi = d.get("name")
                ort =d.get("orthology")
                opat = d.get("pathway")
                ec_num = d.get("enzyme")
                if ort == None:
                    try:
                        ecinf = s.get('ec:'+ec_num)
                        ecparsed = w.parseReaction(ecinf)
                        ort = "kkk"
                        pp = ecparsed.get("pathway")
                        gen_ort =ecparsed.get("orthology")
                        gen_ort = re.sub(":': '",": ",str(gen_ort))
                        gen_ort = re.sub("'","",str(gen_ort))
                        gen_ort = re.sub("{","",str(gen_ort))
                        gen_ort = re.sub("}","",str(gen_ort))
                        gen_ort = gen_ort.split(",")
                    except:
                        pass
                if ort != None:
                    ort = str(ort)
                    ort2 = re.findall('(K\d{5})',ort)
                    hh = 0
                    if ort2 == []:
                        ort2 = "1"
                    for obj in ort2:
                        if ort2 != "1":
                            kinf = s.get(obj)
                            k = w.parseOrthology(kinf)
                            pp = k.get('pathway')
                            genes = k.get("genes")
                        else:
                            genes = gen_ort
                        for objs in genes:
                            for item_name,item_search in zip(strep_names,gene_search):
                                item_name_lc = item_name.lower()
                                if item_name in objs:
                                    hh = 1
                                    gen = re.findall(item_search,objs)
                                    for g in gen:
                                        gaa = s.get(item_name_lc+g,'aaseq')
                                        gaa = gaa+'\n'
                                        info = str(pp)+'\t'+str(g)+'\t'+str(item)+'\t'+str
                                            (defi)+'\t'+str(rxn_words)+'\t'+str(rxn_comp)+'
                                            \t'+str(ec_num)+'\t'+str(opat)+'\t'+str(obj)+'\
                                            t'+str(ort)+'\n'
                                        self.text.insert(END,gaa)
                                        self.text2.insert(END,info)
                    if hh == 1:
                        pass
                    elif hh == 0:
                        gaa = "there are not Streptomyces genes associated with this
                            reaction"+'\n'
                        info = str(opat)+'\t'+'No info'+'\t'+str(item)+'\t'+str(defi)+'\t'
                            +str(rxn_words)+'\t'+str(rxn_comp)+'\t'+str(ec_num)+'\t'+str(
                            opat)+'\t'+str(obj)+'\t'+str(ort)+'\n'
                        self.text.insert(END,gaa)
                        self.text2.insert(END,info)
                elif ort == None:
                    gaa = "The reaction does not have an orthology number"+'\n'
                    try:
                        info = str(opat)+'\t'+'No info'+'\t'+str(item)+'\t'+str(defi)+'\t'
                            +str(rxn_words)+'\t'+str(rxn_comp)+'\t'+str(ec_num)+'\t'+str(
                            opat)+'\t'+'No orthology'+'\t'+'No orthology'+'\n'
                    except:
                        info = 'No info'+'\t'+'No info'+'\t'+str(item)+'\t'+str(defi)+'\t'
                            +str(rxn_words)+'\t'+str(rxn_comp)+'\t'+str(ec_num)+'\t'+'No
```

```
                        info'+'\t'+'No orthology'+'\t'+'No orthology'+'\n'
                self.text.insert(END,gaa)
                self.text2.insert(END,info)
        except IndexError:
            gaa = "Something went wrong! Check the input"
            info = "Something went wrong! Check the input"
            self.text.delete(0.0, END)
            self.text.insert(END,gaa) ##0.0 possition  row,column
            self.text2.delete(0.0, END)
            self.text2.insert(END,info) ##0.0 possition  row,column
```

F.3.2 Perform Blast and sort the output file

```
def performBlast(self):
    from Bio.Blast.Applications import NcbiblastpCommandline
    from Bio.Blast import NCBIWWW
    from Bio.Blast import NCBIXML
    from Bio import SeqIO
    from Bio.Seq import Seq
    from collections import namedtuple
    from operator import itemgetter
    from itertools import groupby

    self.text3.delete(0.0, END)
    self.text4.delete(0.0, END)
    self.text5.delete(0.0, END)

    identity_per = self.favorite.get() ##get the input from the gui
    blastp_cline = NcbiblastpCommandline(query=name1+".fasta", db="aa_sle_2015",
        outfmt=5,
                                         max_target_seqs=10, out=name1+"_blastout.xml"
                                             )
    stdout, stderr = blastp_cline()

    def parse_hit(s):
        try:
            querydef = record.query
            hitid = record.alignments[j].hit_id
            hitdef = record.alignments[j].hit_def
            hspevalue = record.alignments[j].hsps[0].expect
            identity = (record.alignments[j].hsps[0].identities)*100/(record.
                alignments[j].hsps[0].align_length)
            fi.write(str(querydef)+'\t'+str(hitid)+'\t'+str(hitdef)+'\t'+str(identity)
                +'\t'+str(hspevalue)+'\n')
        except IndexError:
            pass

    ##---Parse output of Blast---####
    with open(name1+"_blastout.txt","w") as fi:
        result_handle = open(name1+"_blastout.xml")
        out_parse = NCBIXML.parse(result_handle)
        for record in out_parse:
            j=0
            for hit in record.alignments:
                parse_hit(hit)
                j+=1
    ##Write the SCO and SLE numbers in a new column
    with open(name1+"_blastout.txt", "r") as f,\
         open(name1+"_blastout_sle.txt", "w") as out:
        lines = [line.split('\t') for line in f]
        for line in lines:
            line[4] = line[4].strip()
            item1 = line[0]
            gen = item1.split(' ')
            item2 = line[2]
            sle = item2.split(' ')
```

```
                out.write(str(gen[0])+'\t'+str(sle[0])+'\t'+str(line[2])+'\t'+str(line[3])
                    +'\t'+str(line[4])+'\n')

    ##Select hits that are over a defined threshold
    with open(name1+"_blastout_sle.txt", "r") as f,\
         open(name1+"_blastout_sle_"+identity_per+"id.txt", "w") as out:
        out.write('\t'+'\t'+'\t'+'\t'+'\n') ## incorporar una linea en blanco para que
             no haya problemas cuando se ordena
        lines = [line.split('\t') for line in f]
        for line in lines:
            line[4] = line[4].strip()
            if int(line[3])>float(identity_per) and float(line[4])<0.001:
                out.write(str(line[0])+'\t'+str(line[1])+'\t'+str(line[2])+'\t'+str(
                    line[3])+'\t'+str(line[4])+'\n')

    ##Sort by Sle, Sle number and identity percentage
    with open(name1+"_blastout_sle_"+identity_per+"id.txt", "r") as fe,\
         open(name1+"_blastout_sle_"+identity_per+"id_o.txt", "w") as out:
        rows = [row.split('\t') for row in fe]
        for row in rows:
            rows.sort(key=itemgetter(1,3), reverse=True)
            #rows =sorted(rows, key=itemgetter(2,3), reverse=True)
            out.write('{0}'.format('\t'.join(row)))

    ##Clump all the \textit{Streptomyces} genes that are associated with a Sle gene
        and also show the best hit
    with open(name1+"_blastout_sle_"+identity_per+"id_o.txt", "r") as fe,\
        open(name1+"_blastout_"+identity_per+"id_bestHit.txt", "w") as out1,\
        open(name1+"_blastout_"+identity_per+"id_hitSle.txt", "w") as out2:
        rows = [row.split('\t') for row in fe]
        sle_list = []
        for row in rows:
            if row[0] != "":
                row[4] = row[4].strip()
                sle = row[1]
                if sle not in sle_list:
                    sle_list.append(sle)
                    bhit = row[0]
                    bhits = str(sle)+'\t'+str(bhit)+'\t'+str(row[3])+'\t'+str(row[4])+
                        '\n'
                    self.text3.insert(END,bhits)
                    out1.write(str(row[0])+'\t'+str(row[1])+'\t'+str(row[2])+'\t'+str(
                        row[3])+'\t'+str(row[4])+'\n')
        #sle_list2 = []
        for item in sle_list:
            gen_list = []
            for row in rows:
                if row[0] != "":
                    row[1] = row[1].strip()
                    row[4] = row[4].strip()
                    gen = row[0]
                    if row[1] == item:
                        gen_list.append(gen)
            hitsle = str(item)+'\t'+str(gen_list)+'\n'
            self.text4.insert(END,hitsle)
            out2.write(str(gen_list)+'\t'+str(item)+'\n')

    with open(name1+"_blastout_sle_"+identity_per+"id_o.txt", "r") as fe,\
        open(name2+"2.txt", "r") as f2,\
        open(name2+"2_with_sle_"+identity_per+"id_hitSle.txt", "w") as out,\
        open("temp.txt", "w") as out2:
        lines = [line.split('\t') for line in f2]
        rows = [row.split('\t') for row in fe]
        reac_list = []
        for line in lines:
            genin = line[1] ##streptomyces genes
            line[9] = line[9].strip()
            a = 0
            for row in rows:
                ggg = row[0]
                ppp = re.findall(":(.+)",ggg)
```

```
                    ppp = str(ppp)
                    ppp = re.sub("\['","",ppp)
                    ppp = re.sub("'\]","",ppp)
                    sle = row[1]
                    if genin in ppp:
                        a = 1
                        out.write(str(line[0])+'\t'+str(sle)+'\t'+str(line[1])+'\t'+str(
                            line[2])+'\t'+str(line[3])+'\t'
                             +str(line[4])+'\t'+str(line[5])+'\t'+str(line[6])+'\t'+str(
                                 line[7])+'\t'+str(line[8])+'\t'
                             +str(line[9])+'\n')
                if a == 0:
                    out.write(str(line[0])+'\t'+'GAP'+'\t'+str(line[1])+'\t'+str(line[2])+
                        '\t'+str(line[3])+'\t'
                             +str(line[4])+'\t'+str(line[5])+'\t'+str(line[6])+'\t'+str(
                                 line[7])+'\t'+str(line[8])+'\t'
                             +str(line[9])+'\n')
                elif a == 1:
                    pass
                reac = line[2]
                if reac not in reac_list:
                    reac_list.append(reac)
                    out2.write(str(line[0])+'\t'+str(line[1])+'\t'+str(line[2])+'\t'+str(
                        line[3])+'\t'
                             +str(line[4])+'\t'+str(line[5])+'\t'+str(line[6])+'\t'+str(
                                 line[7])+'\t'+str(line[8])+'\t'
                             +str(line[9])+'\n')

        ##Create a file with the clumped genes sorted by reactions
        gen_info = namedtuple("gene_information", ["Sle_genes", "Sco_genes", "reaction_num
            "])
        rt2 = namedtuple("GENandEC", ["Strep_number","EC_number"])
        rt3 = namedtuple("GENandEC", ["SLE_number","EC_number"])
        with open(name2+"2_with_sle_"+identity_per+"id_hitSle.txt", "r") as fe,\
            open(name2+"2_with_sle_"+identity_per+"id_hitSle_by_rx.txt", "w") as out,\
            open("C:\\Users\\Valeria\\Dropbox\\Thesis Streptomyces files\\python _2014\\
                deltaG.txt") as delt,\
            open("151123_abbr_compounds.txt","r") as mm,\
            open("temp.txt", "r") as f2:
            rows = [row.split('\t') for row in fe]
            lines = [line.split('\t') for line in f2]
            lineas = [linea.split('\t') for linea in delt]
            mets = [met.split('\t') for met in mm]
            rx_list=[]
            rx_list2=[]
            tup_list = []
            for row in rows:
                row[10] = row[10].strip()
                if row[3] not in rx_list:
                    rx_list.append(row[3])
            for item in rx_list:
                genin_list=[]
                genin_list2=[]
                slein_list=[]
                jj=0
                for row in rows:
                    row[10] = row[10].strip()
                    slein = row[1]
                    genin = row[2]
                    rx = row[3]
                    if item == rx and slein not in slein_list and slein != "GAP":
                        jj = 1
                        slein_list.append(slein)
                        genin_list.append(genin)
                    elif item == rx and slein == "GAP" and genin not in genin_list2:
                        genin_list2.append(genin)
                if jj == 1:
                    tup = gen_info(Sle_genes = slein_list, Sco_genes = genin_list,
                        reaction_num = item)
                elif jj == 0:
```

```
                    tup = gen_info(Sle_genes = "GAP", Sco_genes = genin_list2,
                        reaction_num = item)
                tup_list.append(tup)
            for line,inftup in zip(lines,tup_list):
                line[9] = line[9].strip()
                rx = line[2]
                kn = line[8]
                ecline = re.findall(kn+"(.+?)\]",line[9])
                ecline = str(ecline)
                ec = re.findall("(EC:.+)",ecline)
                ec = str(ec)
                ec = re.sub("\[","", ec)
                ec = re.sub("\]","", ec)
                ec = re.sub("'","", ec)
                ec = re.sub('"',"", ec)
                sl2_list = []
                sl2_list2 = []
                sl2_list3 = []
                ringen1_list = []
                ss = 0
                pp = 0
                for row in rows:
                    row[10] = row[10].strip()
                    rx2 = row[3]
                    kn2 = row[9]
                    sl2 = row[1]
                    ecline2 = re.findall(kn2+"(.+?)\]",row[10])
                    ecline2 = str(ecline2)
                    ec2 = re.findall("(EC:.+)",ecline2)
                    ec2 = str(ec2)
                    ec2 = re.sub("\[","", ec2)
                    ec2 = re.sub("\]","", ec2)
                    ec2 = re.sub("'","", ec2)
                    ec2 = re.sub('"',"", ec2)
                    if rx == rx2 and sl2 not in sl2_list and sl2 != "GAP":
                        sl2_list.append(sl2)
                        if ec2 == ec:
                            ss = 1
                            sl2_list2.append(sl2)
                            ringen2 = rt3(SLE_number = sl2_list2, EC_number = ec2)
                        elif ec2 != ec:
                            pp = 1
                            sl2_list3.append(sl2)
                            ringen1 = rt3(SLE_number = sl2_list3, EC_number = ec2)
                if ss == 1:
                    ringen1_list.append(ringen2)
                if pp == 1:
                    ringen1_list.append(ringen1)
                deltaG = "No info"
                for linea in lineas:
                    linea[1]= linea[1].strip()
                    if rx in linea:
                        j = float(linea[1])
                        if float(linea[1]) < 0:
                            line[4] = re.sub('<=>', '=>',line[4])
                            line[5] = re.sub('<=>', '=>',line[5])
                            deltaG = j
                        elif float(linea[1]) > 0 and float(linea[1]) != 12345:
                            line[4] = re.sub('<=>', '<=',line[4])
                            line[5] = re.sub('<=>', '<=',line[5])
                            deltaG = j
                        elif float(linea[1]) == 0:
                            deltaG = j
                        elif float(linea[1]) == 12345:
                            deltaG = "No delta G"
                c_abs = line[5]
                comps = c_abs.split(" ")
                for item in comps:
                    for met in mets:
                        cname = met[0]
                        if item == cname:
```

```
                        c_abs = re.sub(item+'\s',met[1]+' ',c_abs)
                        c_abs = re.sub(item+'\(n\)',met[1]+'(n)',c_abs)
                        c_abs = re.sub(item+'\(n\+2\)',met[1]+'(n+2)',c_abs)
                        c_abs = re.sub(item+'$',met[1],c_abs)
            reactions_sle = str(line[0])+'\t'+str(inftup.Sco_genes)+'\t'+'-'+'\t'+str(
                inftup.Sle_genes)+'\t'+str(ringen1_list)+'\t'+str(inftup.reaction_num)+
                '\t'+str(line[3])+'\t'+str(c_abs)+'\t'+str(line[4])+'\t'+str(line[5])+'
                \t'+str(line[6])+'\t'+str(line[7])+'\t'+str(line[8])+'\t'+str(deltaG)+'
                \n'
            self.text5.insert(END,reactions_sle)
            out.write(str(line[0])+'\t'+str(inftup.Sco_genes)+'\t'+'-'+'\t'+str(inftup
                .Sle_genes)+'\t'+str(ringen1_list)
                        +'\t'+str(inftup.reaction_num)+'\t'+str(line[3])+'\t'+str(c_abs)
                            +'\t'+str(line[4])+'\t'
                        +str(line[5])+'\t'+str(line[6])+'\t'+str(line[7])+'\t'+str(line
                            [8])+'\t'+str(deltaG)+'\n')
```

F.3.3 Get compound information

```
def retrieve_comp(self):
    from bioservices import Kegg
    from bioservices import *
    s = Kegg()
    w = KeggParser()
    import re
    content = self.textinput.get(0.0, END)
    content = str(content)
    self.text.delete(0.0, END)
    compound_list = re.findall("(C\d{5})",content)
    compound_list = sorted(set(compound_list))
    for compound in compound_list:
        cc  = w.get(compound)
        pubchem = re.findall('PubChem:\s(\d+)',cc)
        CAS = re.findall('CAS:\s(\d+)',cc)
        ChEBI = re.findall('ChEBI:\s(\d+)',cc)
        ccc = w.parseCompound(cc)
        name = ccc.get('name')
        name = str(name)
        name = name.split(', ')
        pnom = name[0]
        pnom = re.sub("\['","",pnom)
        pnom = re.sub(";'","",pnom)
        pnom = re.sub('\["','',pnom)
        pnom = re.sub(';"','',pnom)
        form = ccc.get('formula')
        otherdb = ccc.get('dblinks')
        molw = ccc.get('mol_weight')
        exmass = ccc.get('exact_mass')
        info = str(compound)+'\t'+str(pnom)+'\t'+str(form)+'\t'+str(pubchem)+'\t'+str(
            CAS)+'\t'+str(ChEBI)+'\n'
        self.text.insert(END,info)
```

F.3.4 Abbreviate compounds name

```
def abbr_comp(self):
    from collections import namedtuple
    from operator import itemgetter
    from itertools import groupby
    import re
    self.text2.delete(0.0,END)
    c=0
    tulist=[]
    tuexlist=[]
```

```
    cmm_n = [r"Oxygen",r"Orthophosphate",r"Diphosphate",r"Ammonia",r"diphosphate",r"bisphosphate",r"phosphate",
             r"Phospho",r"Phospha",r"Diphospho",r"ethanolamine",r"semialdehyde",r"acetaldehyde",r"aldehyde",
             r"acid",r"alpha",r"beta",r"Acyl-carrier-protein",r"Acyl-carrier protein",r"acyl-carrier protein",
             r"acyl-carrier-protein",r"\[acp\]",r"pimeloyl",r"glutaryl",r"oxo",r"Oxo",r"Dimethyl",r"methyl",
             r"Methyl",r"Hydrogen peroxide",r"Hydroxy",r"Hydro",r"Acetoacetyl",r"acetoacetyl",r"Acetyl",
             r"Amino",r"tetrahydrofolate",r"tetrahydro",r"Tetra",r"Dehydro",r"ribosyl",r"glycin",r"amide",
             r"acetylmuramoyl",r"glucuronate","Deoxyadenosine",r"Deoxy",r"Formyl",'Glutamate','ubiquinone',
             'thionate','Propionyl','Phenyl','decanoyl','enolpyruvate','lactone','Hquinone','iso','Iso']
    cmm_ab = ['02',"pi","ppi","NH3","dp","bp","p","P","P","Dp","etholamne","salde","acetalde","alde","ad","a","b","acp","ACP",
              "acp","acp","-acp","pmoyl","glryl","o","O-","dmyl-","myl-","Myl-","H202","Hxy","H","Actactyl","actactyl",
              "Acetyl-","Ami","t-h-folate","th","T","Dh","rbsyl","gly","ade","actylmuoyl","glunate","Dx-adenosine","Dx",
              "Fryl","Gluta","-ubiqnone","-thionate","Propionyl-","Phenyl-","-decanoyl","enol-pyruvate","-lactone",
              "H-quinone","-iso-","-Iso-"]
    ab_sub = ['", ','"',"', ","'","\[","\]","\(","\)"]
    tu = namedtuple("Comp_abr", ["compound_n", "abrev","name"])
    with open(name2+"2.txt","r") as f,\
         open(name2+"2_abb.txt","w") as out:
        lines = [line.split('\t') for line in f]
        for line in lines:
            abr = 'sinabrev'
            line[5]=line[5].strip()
            comp2 = line[1]
            cnum = line[0]
            ex = re.findall("ex",cnum)
            for item_n,item_ab in zip(cmm_n,cmm_ab):
                comp2 = re.sub(item_n,item_ab,comp2)
            leco = len(comp2)
            if ex == []:
                if leco <= 5:
                    abr = comp2
                    atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                    if abr not in alist:
                        alist.append(abr)
                        tulist.append(atup)
                elif leco > 5:
                    ff = comp2.split(' ')
                    dd=[]
                    qq=[]
                    ee=[]
                    tt=[]
                    if len(ff) == 1:
                        rr = comp2.split('-')
                        for item in rr:
                            aa = []
                            w = 0
                            for obj in tulist:
                                if item == obj.name:
                                    item1 = item2 = item3 = item4 = obj.abrev
                                    w = 1
                            if w == 0:
                                if '1' in item or '2' in item or '3' in item or '4' in item or '5' in item or 'tRNA(' in item:
                                    item1 = item2 = item3 = item4 = item
                                elif 'oyl' in item:
                                    item1 = item[:3]+'oyl'; item2 = item[:4]+'oyl'; item3 = item[:5]+'oyl'; item4 = item
                                elif 'nyl' in item:
                                    item1 = item[:3]+'nyl'; item2 = item[:4]+'nyl'; item3 = item[:5]+'nyl'; item4 = item
```

```
                                    elif 'alde' in item:
                                        item1 = item[:3]+'lde'; item2 = item[:4]+'lde';
                                            item3 = item[:5]+'lde'; item4 = item
                                    elif 'benzoate' in item:
                                        item1 = item[:3]+'bzte'; item2 = item[:4]+'bzte';
                                            item3 = item[:5]+'bzte'; item4 = item
                                    elif 'nal' in item:
                                        item1 = item[:3]+'nal'; item2 = item[:4]+'nal';
                                            item3 = item[:5]+'nal'; item4 = item
                                    elif 'nol' in item:
                                        item1 = item[:3]+'nol'; item2 = item[:4]+'nol';
                                            item3 = item[:5]+'nol'; item4 = item
                                    elif 'oate' in item:
                                        item1 = item[:3]+'te'; item2 = item[:4]+'te';
                                            item3 = item[:5]+'te'; item4 = item
                                    else:
                                        item1 = item[:3]; item2 = item[:4]; item3 = item
                                            [:5]; item4 = item
                                dd.append(item1)
                                qq.append(item2)
                                ee.append(item3)
                                tt.append(item4)
                            abr = str(dd)
                            for item_sub in ab_sub:
                                abr = re.sub(item_sub,"",abr)
                            if abr not in alist:
                                alist.append(abr)
                                atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                tulist.append(atup)
                            elif abr in alist:
                                abr = str(qq)
                                for item_sub in ab_sub:
                                    abr = re.sub(item_sub,"",abr)
                                if abr not in alist:
                                    alist.append(abr)
                                    atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                    tulist.append(atup)
                                elif abr in alist:
                                    abr = str(ee)
                                    for item_sub in ab_sub:
                                        abr = re.sub(item_sub,"",abr)
                                    if abr not in alist:
                                        alist.append(abr)
                                        atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                        tulist.append(atup)
                                    elif abr in alist:
                                        abr = str(tt)
                                        for item_sub in ab_sub:
                                            abr = re.sub(item_sub,"",abr)
                                        if abr not in alist:
                                            alist.append(abr)
                                            atup = tu(compound_n=cnum,abrev=abr,name=comp2
                                                )
                                            tulist.append(atup)
                                        elif abr in alist:
                                            print 'It was not possible to abbreviate the
                                                compound:'+str(comp2)
                        elif len(ff) > 1:
                            for item in ff:
                                aa = []
                                z = 0
                                for obj in tulist:
                                    if item == obj.name:
                                        item =obj.abrev
                                        z = 1
                                pp = item.split('-')
                                for ob in pp:
                                    if '1' in ob or '2' in ob or '3' in ob or '4' in ob or
                                         '5' in ob:
                                        dd.append(ob)
                                        qq.append(ob)
```

```
                                    else:
                                        dd.append(ob[:4])
                                        qq.append(ob[:5])
                                abr = str(dd)
                                for item_sub in ab_sub:
                                    abr = re.sub(item_sub,"",abr)
                                if abr not in alist:
                                    alist.append(abr)
                                    atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                    tulist.append(atup)
                                elif abr in alist:
                                    abr = str(qq)
                                    for item_sub in ab_sub:
                                        abr = re.sub(item_sub,"",abr)
                                    if abr not in alist:
                                        alist.append(abr)
                                        atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                        tulist.append(atup)
                                    elif abr in alist:
                                        ii=[]
                                        oo = comp2.split(' ')
                                        for item in oo:
                                            if 'D-' not in item:
                                                item = item[:3]
                                            elif 'D-' in item:
                                                item = re.sub("D-","D",item)
                                                item = item[:4]
                                            ii.append(item)
                                        abr = str(ii)
                                        for item_sub in ab_sub:
                                            abr = re.sub(item_sub,"",abr)
                                        if abr not in alist:
                                            alist.append(abr)
                                            atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                            tulist.append(atup)
                                        elif abr in alist:
                                            print 'It was not possible to abbreviate the
                                                compound:'+str(comp2)
                    else:
                        print 'miss'
                elif ex != []:
                    for obj in tulist:
                        info = obj
                        cnum = str(cnum)
                        cnum = re.sub("ex","",cnum) ##eliminar ex para poder comparar con
                            los compuestos que ya estan abreviados
                        if cnum in info.compound_n:
                            abr = info.abrev+'_ex'
                            if abr not in alist:
                                alist.append(abr)
                                atup = tu(compound_n=cnum,abrev=abr,name=comp2)
                                tuexlist.append(atup)
                out.write(str(line[0])+'\t'+str(abr)+'\t'+str(line[1])+'\t'+str(line[2])+'
                    \t'+str(line[3])+'\t'+
                          str(line[4])+'\t'+str(line[5])+'\n')

    abrlist=[]
    with open(name2+"2_abb.txt","r") as f,\
         open(name2+"2_abb_final.txt","w") as out:
        lines = [line.split('\t') for line in f]
        abrev_num = [">","^10","^1","^2","^3","^4","^5","^6","^7","^8","^9","^\d{1}","
            ^\d{2}",",","\+","-","\/"]
        abrev_new = ["","d","u","d","t","q","c","s","ste","o","n","","","","","",""]
        for line in lines:
            abr = 'sinabrev'
            line[6]=line[6].strip()
            abrev = line[1]
            cnum = line[0]
            for obj_num,obj_new in zip(abrev_num,abrev_new):
                abrev = re.sub(obj_num,obj_new,abrev)
            abrev = re.sub(abrev, abrev+'_c', abrev)
```

```
                abrlist.append(abrev)
                ab_info = str(abrev)+'\t'+str(line[2])+'\n'
                self.text2.insert(END,ab_info)
                out.write(str(line[0])+'\t'+str(abrev)+'\t'+str(line[2])+'\t'+str(line[3])
                    +'\t'+str(line[4])+'\t'+
                          str(line[5])+'\t'+str(line[6])+'\n')
        for item in abrlist:
            if abrlist.count(item) > 1:
                print item
```

F.3.5 Write reactions in COBRApy

```
def write_reactions(self):
    """Write the reactions in Cobrapy format"""
    global exch_list
    import re
    i = 0
    r = {}
    self.text6.delete(0.0,END)
    self.text7.delete(0.0,END)
    self.text8.delete(0.0,END)
    self.text9.delete(0.0,END)
    with open(filename) as f:
        lines = [line.split('\t') for line in f]
        for line in lines:
            i+=1
            reaction_number = line[5]
            item = line[7]
            item = re.sub(r"^(.)",r" \1",item)
            try:
                rec,prod = item.split('<=>')
            except ValueError:
                try:
                    rec,prod = item.split('=>')
                except ValueError:
                    prod,rec = item.split('<=')

            comps = re.findall("(\w*)",rec)
            comps2 = re.findall("(\w*)",prod)
            for item in comps:
                d=0
                try:
                    float(item)
                    d=1
                except ValueError:
                    pass
                if item != '' and d == 0:
                    e = re.findall("(\d)\s"+item,rec)
                    if e == []:
                        rec = re.sub(r"(\s"+item+")",r" 1 \1",rec)
                        #rec = re.sub(r"^("+item+")",r" 1 \1",rec)
            for item in comps2:
                d=0
                try:
                    float(item)
                    d=1
                except ValueError:
                    pass
                if item != '' and d == 0:
                    e = re.findall("(\d)\s"+item,prod)
                    if e == []:
                        prod = re.sub(r"(\s"+item+")",r" 1 \1",prod)
                        #prod = re.sub(r"^("+item+")",r" 1 \1",prod)
            cmsr = rec.split(' ')
            estr = []
            react = []
            for item in cmsr:
```

```
                    if item != "" and item != "+":
                        try:
                            float(item)
                            estr.append(item)
                        except ValueError:
                            react.append(item)
                cmsp = prod.split(' ')
                estp = []
                product = []
                for item in cmsp:
                    if item != "" and item != "+":
                        try:
                            float(item)
                            estp.append(item)
                        except ValueError:
                            product.append(item)
                cob = []
                for er,r in zip(estr,react):
                    er = float(er)*-1
                    an = str(r)+': '+str(er)
                    cob.append(an)
                for ep,p in zip(estp,product):
                    ep = float(ep)
                    an = str(p)+': '+str(ep)
                    cob.append(an)
                cob = str(cob)
                cob = re.sub("\[","",cob)
                cob = re.sub("\]","",cob)
                cob = re.sub("'","",cob)
                add_met_to_reac = 'r['+str(i)+'].add_metabolites({'+str(cob)+'})'+' # '+
                    str(reaction_number)+'\n'
                add_reac = 'cobra_model.add_reaction('+'r['+str(i)+'])'+'\n'
                self.text6.insert(END,add_reac)
                self.text7.insert(END,add_met_to_reac)
        with open(metfile) as f2:
            lines = [line.split('\t') for line in f2]
            i= 0
            e = {}
            exch_list = []
            for line in lines:
                cnum = line[0]
                met = line[1]
                name = line[2]
                name = re.sub('"',"'", name)
                name = re.sub("\['","", name)
                name = re.sub(";',.*\]","", name)
                if 'ex' in cnum:
                    exch_list.append(met)
                    m_info = met+' = '+'Metabolite("'+str(met)+'", formula="'+str(line[3])
                        +'", name="'+str(name)+'", compartment="'+'e")'+' ## '+str(cnum)+'\
                        n'
                    self.text8.insert(END, m_info)
                    i+=1
                    exch = 'e['+str(i)+'].add_metabolites({'+met+': -1}) ## Exchange_'+met
                        +'\n'
                    add_exch = 'cobra_model.add_reaction(e['+str(i)+'])'+'\n'
                    self.text9.insert(END, exch)
                    self.text9.insert(END, add_exch)

                elif 'ex' not in cnum:
                    m_info = met+' = '+'Metabolite("'+str(met)+'", formula="'+str(line[3])
                        +'", name="'+str(name)+'", compartment="'+'c")'+' ## '+str(cnum)+'\
                        n'
                    self.text8.insert(END, m_info)
```

F.3.6 To create a python module that would contain the information to be loaded when creating the GSM

```
def saveinC(self):
    """Save the reactions and metabolites in cobrapy format into a file"""
    global modelname
    modelname = self.inputs3.get()
    f1 = self.text8.get(0.0,END)
    mets = [met.split('\n') for met in f1]
    f2 = self.text7.get(0.0,END)
    reactions = [reaction.split('\n') for reaction in f2]
    f3 = self.text6.get(0.0,END)
    addreactions = [addreaction.split('\n') for addreaction in f3]
    f4 = self.text9.get(0.0, END)
    exchange_rxns = [exch.split('\n') for exch in f4]
    with open("info_cobrapy"+'temp.txt', "w") as out:
        out.write("import re"+'\n')
        out.write("import numpy as np"+'\n')
        out.write("from cobra import Model, Reaction, Metabolite"+'\n')
        out.write("cobra_model = Model('"+modelname+"')"+'\n'+'\n')
        out.write(f1+'\n')
        out.write('\n'+'r = {}'+'\n')
        out.write('ic = 0'+'\n')
        out.write("with open('"+filename+"') as f:"+'\n')
        out.write('\t'+"lines = [line.split(\'\\t\') for line in f]"+'\n')
        out.write('\t'+"for line in lines:"+'\n')
        out.write('\t'+'\t'+"ic+=1"+'\n')
        out.write('\t'+'\t'+"reaction_number = line[5]"+'\n')
        out.write('\t'+'\t'+"r[ic] = Reaction(reaction_number)"+'\n')
        out.write('\t'+'\t'+"path = line[0]"+'\n')
        out.write('\t'+'\t'+"r[ic].name = str(line[6])"+'\n')
        out.write('\t'+'\t'+"r[ic].subsystem = path"+'\n')
        out.write('\t'+'\t'+"sle = line[3]"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'" ",'+"'',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'"'+"'"+'"'+",'',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+"'"+"\["+"'"+",'',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'"'+"\]"+'"'+",'',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'"or"'+",' or ',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'"and"'+",' and ',sle)"+'\n')
        out.write('\t'+'\t'+"sle = re.sub("+'",'"'+",' or ',sle)"+'\n')
        out.write('\t'+'\t'+"if sle != 'GAP':"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].gene_reaction_rule = sle"+'\n')
        out.write('\t'+'\t'+"if '<=>' in line[7]:"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].lower_bound = -1000"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].upper_bound = 1000"+'\n')
        out.write('\t'+'\t'+"elif '<= ' in line[7]:"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].lower_bound = -1000"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].upper_bound = 0"+'\n')
        out.write('\t'+'\t'+"elif ' =>' in line[7]:"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].lower_bound = 0"+'\n')
        out.write('\t'+'\t'+'\t'+"r[ic].upper_bound = 1000"+'\n')

        out.write('\n')
        out.write(f2+'\n')
        out.write('\n')
        out.write(f3+'\n')

        out.write('\n'+'e = {}'+'\n')
        out.write('ii = 0'+'\n')
        out.write("with open('"+metfile+"') as f2:"+'\n')
        out.write('\t'+"lines = [line.split(\'\\t\') for line in f2]"+'\n')
        out.write('\t'+"for line in lines:"+'\n')
        out.write('\t'+'\t'+"cnum = line[0]"+'\n')
        out.write('\t'+'\t'+"met = line[1]"+'\n')
        out.write('\t'+'\t'+"if 'ex' in cnum:"+'\n')
        out.write('\t'+'\t'+'\t'+"ii+=1"+'\n')
        out.write('\t'+'\t'+'\t'+"e[ii] = Reaction('Ex'+met)"+'\n')
        out.write('\t'+'\t'+'\t'+"e[ii].name = 'Exchange_'+met"+'\n')
        out.write('\t'+'\t'+'\t'+"e[ii].lower_bound = -1000"+'\n')
        out.write('\t'+'\t'+'\t'+"e[ii].upper_bound = 1000"+'\n')
```

```
68
69          out.write('\n')
70          out.write(f4+'\n')
71          out.write("from cobra.io import write_sbml_model"+'\n')
72          out.write("sbml_out_file = '"+modelname+".xml'"+'\n')
73          out.write("write_sbml_model(cobra_model, sbml_out_file)")
74
75      with open("info_cobrapy"+'temp.txt',"r") as f,\
76          open("info_cobrapy"+'.py',"w") as out:
77          lines = [line.split('\t') for line in f]
78          out.write('def writemodel(self):'+'\n')
79          for line in lines:
80              out.write('\t'+'{0}'.format('\t'.join(line)))
```

F.3.7 Create the model

```
1   def createmodelSLE(self):
2       global cobra_model
3       from cobra.io import read_sbml_model, write_sbml_model
4       import info_cobrapy
5       reload(info_cobrapy)
6       from info_cobrapy import writemodel
7       writemodel(self)
8       self.entry.delete(0,END)
9       self.entry.insert(0,modelname+".xml")
10      self.text.delete(0.0,END)
11      cobra_model = read_sbml_model(modelname+'.xml')
12      self.text.insert(END,'%i reaction' % len(cobra_model.reactions)+'\n')
13      self.text.insert(END,'%i metabolites' % len(cobra_model.metabolites)+'\n')
14      self.text.insert(END, '%i genes' % len(cobra_model.genes)+'\n')
```

F.3.8 Check mass balance

```
1   def massbalancecheck(self):
2       """Check mass balance"""
3       from cobra import Model, Reaction, Metabolite
4       import re
5       import sys
6       sys.dont_write_bytecode = True
7       mets_info = self.listbox2.get(0, END)
8       self.text3.delete(0.0,END)
9       with open("massbalance.py","w") as out:
10          out.write('def checkmassbalance(self):'+'\n')
11          out.write('\t'+"from cobra import Model, Reaction, Metabolite"+'\n')
12          out.write('\t'+"from cobra.io import read_sbml_model, write_sbml_model"+'\n')
13          #out.write('\t'+'global b'+'\n')
14          out.write('\t'+"cobra_model = Model('checkmb')"+'\n')
15          for m in mets_info:
16              line = m.split(' ')
17              met = line[0]
18              form = line[1]
19              if '_e' in met:
20                  m_info = met+' = '+'Metabolite("'+str(met)+'", formula="'+str(line[1])
                        +'", name="'+str(met)+'", compartment="'+'e")'+'\n'
21                  out.write('\t'+m_info)
22                  #self.text3.insert(END, m_info)
23              elif '_e' not in met:
24                  m_info = met+' = '+'Metabolite("'+str(met)+'", formula="'+str(line[1])
                        +'", name="'+str(met)+'", compartment="'+'c")'+'\n'
25                  #self.text3.insert(END, m_info)
26                  out.write('\t'+m_info)
27          item = map(int, self.listbox.curselection())
28          value = self.listbox.get(item[0])
```

```
            mmet, rinfo = value.split("in:")
            item = re.sub(r"^(.)",r" \1",rinfo)
            reaction_number = "check_p1"
            p1 = Reaction(reaction_number)
            out.write('\t'+"reaction_number = 'check_p1'"+'\n')
            out.write('\t'+"p1 = Reaction(reaction_number)"+'\n')
            try:
                rec,prod = item.split('<=>')
            except ValueError:
                try:
                    rec,prod = item.split('=>')
                except ValueError:
                    prod,rec = item.split('<=')
            comps = re.findall("(\w*)",rec)
            comps2 = re.findall("(\w*)",prod)
            for item in comps:
                d=0
                try:
                    float(item)
                    d=1
                except ValueError:
                    pass
                if item != '' and d == 0:
                    e = re.findall("(\d)\s"+item,rec)
                    if e == []:
                        rec = re.sub(r"(\s"+item+")",r" 1 \1",rec)
            for item in comps2:
                d=0
                try:
                    float(item)
                    d=1
                except ValueError:
                    pass
                if item != '' and d == 0:
                    e = re.findall("(\d)\s"+item,prod)
                    if e == []:
                        prod = re.sub(r"(\s"+item+")",r" 1 \1",prod)
            cmsr = rec.split(' ')
            estr = []
            react = []
            for item in cmsr:
                if item != "" and item != "+":
                    try:
                        float(item)
                        estr.append(item)
                    except ValueError:
                        react.append(item)
            cmsp = prod.split(' ')
            estp = []
            product = []
            for item in cmsp:
                if item != "" and item != "+":
                    try:
                        float(item)
                        estp.append(item)
                    except ValueError:
                        product.append(item)
            cob = []
            for er,r in zip(estr,react):
                er = float(er)*-1
                an = str(r)+': '+str(er)
                cob.append(an)
            for ep,p in zip(estp,product):
                ep = float(ep)
                an = str(p)+': '+str(ep)
                cob.append(an)
            cob = str(cob)
            cob = re.sub("\[","",cob)
            cob = re.sub("\]","",cob)
            cob = re.sub("'","",cob)
```

```
            add_met_to_reac = 'p1.add_metabolites({'+str(cob)+'})'+' # '+str(
                reaction_number)+'\n'
            add_reac = 'cobra_model.add_reaction(p1)'+'\n'
            out.write('\t'+'try:'+'\n')
            out.write('\t'+'\t'+add_met_to_reac)
            out.write('\t'+'\t'+add_reac)
            out.write('\t'+'\t'+'b = p1.check_mass_balance()'+'\n')
            out.write('\t'+'\t'+"self.text3.insert(0.0, b)"+'\n')
            out.write('\t'+'except Exception, err:'+'\n')
            out.write('\t'+'\t'+'self.text3.insert(0.0, err)'+'\n')
        import massbalance
        reload(massbalance)
        from massbalance import checkmassbalance
        checkmassbalance(self)
```

F.3.9 Balance protons or water

```
def pbalance(self):
    self.text5.delete(0.0,END)
    selection = self.listbox.curselection()
    for item in range(0,len(selection)):
        value = self.listbox.get(selection[item])
        rn, b = value.split(" is unbalanced: ")
        rn2 = cobra_model.reactions.get_by_id(rn)
        d = re.findall("'.': .*?,|'.': .*?\}", b)
        z = re.findall("'(.)': .*?[,|\}]", b)
        f = re.findall("'.': (.*?)[,|\}]", b)
        new_dict = {}
        for c,w in zip(f,z):
            c = float(c)
            new_dict[w] = c
        if 'H' in new_dict and len(new_dict) == 1 and new_dict['H'] > 0:
            va = new_dict['H']
            rn2.add_metabolites({cobra_model.metabolites.get_by_id("H_c"): -va})
            bi = rn2.check_mass_balance()
            if bi == {}:
                inf = "Reaction is balanced"
            elif bi != {}:
                inf = str(bi)
        elif 'H' in new_dict and len(new_dict) == 1 and new_dict['H'] < 0:
            va = new_dict['H']
            va = abs(va)
            rn2.add_metabolites({cobra_model.metabolites.get_by_id("H_c"): va})
            bi = rn2.check_mass_balance()
            if bi == {}:
                inf = "Reaction is balanced"
            elif bi != {}:
                inf = str(bi)
        elif len(new_dict) == 2 and 'H' in new_dict and 'O' in new_dict:
            if new_dict['H'] > 0 and new_dict['O'] > 0:
                va = new_dict['H']
                va2 = new_dict['O']
                if va == 2*va2:
                    rn2.add_metabolites({cobra_model.metabolites.get_by_id("H2O_c"): -
                        va2})
                    bi = rn2.check_mass_balance()
                    if bi == {}:
                        inf = 'Reaction is balanced'
                    elif bi != {}:
                        inf = str(bi)
            elif new_dict['H'] < 0 and new_dict['O'] < 0:
                va = new_dict['H']
                va = abs(va)
                va2 = new_dict['O']
                va2 = abs(va2)
                if va == 2*va2:
```

```
                        rn2.add_metabolites({cobra_model.metabolites.get_by_id("H2O_c"):
                            va2})
                        bi = rn2.check_mass_balance()
                        if bi == {}:
                            inf = 'Reaction is balanced'
                        elif bi != {}:
                            inf = str(bi)
                else:
                    inf = 'Error'
            else:
                inf = "The reaction unbalance involved more than H or H2O"
            self.text5.insert(END, str(rn2)+'\t'+inf+'\n')
```

F.3.10 Simulate growth in complex media

```
def SimulateGrowth_cm(self):
    import cobra
    import cobra.io
    import cobra.test
    self.text3.delete(0.0, END)
    self.text2.delete(0.0, END)
    cobra_model.optimize(solver='gurobi')
    comp1_list = ['ExDGlu_e','ExGlyc_e','ExMal_e','ExDFru_e','ExqAmibzte_e','ExBio_e',
        'ExIno_e','ExFol_e','ExNico_e',
                  'ExRib_e','ExPan_e']
    comp2_list = ['ExLAla_e','ExLArg_e','ExLAsp_e','ExLCys_e','ExGluta_e','ExGly_e','
        ExLHis_e','ExLIsoleu_e','ExLLeu_e',
                  'ExLLys_e','ExLMet_e','ExLPhenylala_e','ExLPro_e','ExLSer_e','
                      ExLThr_e','ExLTry_e','ExLTyr_e','ExLVal_e']
    for item in comp1_list:
        cobra_model.reactions.get_by_id(item).lower_bound = -10
        cobra_model.reactions.get_by_id(item).upper_bound = 1000

    for item in comp2_list:
        cobra_model.reactions.get_by_id(item).lower_bound = -0.1
        cobra_model.reactions.get_by_id(item).upper_bound = 1000

    cobra_model.reactions.get_by_id('ExaaTre_e').lower_bound = -1
    cobra_model.reactions.get_by_id('ExaaTre_e').upper_bound = 1000

    cobra_model.objective = 'b00001' ##Biomass objective function
    cobra_model.optimize(solver='gurobi')
    opi1 = cobra_model.solution.f

    self.text3.insert(END,str(opi1)+'\n')
    self.text3.insert(END, 'Status:'+'\t'+cobra_model.solution.status+'\n')
    dd = cobra_model.solution.x_dict
    for item in dd:
        xsol = str(item)+': '+str(dd[item])
        self.text2.insert(END, xsol+'\n')
```

F.3.11 Apply FSEOF

```
def FSEOF(self):
    import cobra.test
    from cobra.io import read_sbml_model, write_sbml_model
    from collections import namedtuple
    self.text4.delete(0.0, END)
    fseof_target = self.combox1.get()
    ntimes = self.combox2.get()
    cobra_model.optimize(solver='gurobi')

    ##----------------------------------------------------------------------
```

```
    ##1. Calculate vj(initial) by maximizing the biomass formation.
    ##-------------------------------------------------------------------------
    cobra_model.objective = 'ExBiomass_e' ##objective function
    cobra_model.optimize(solver='gurobi')
    v_biomass = cobra_model.solution.f
    vin = cobra_model.solution.x_dict
    tu = namedtuple("v_initial",["rxn_name", "rxn_flux_ini"])
    v_fluxes_ini = []
    for item in vin:
        vj_name = str(item)
        vj_ini = vin[item]
        v_flux_ini = tu(rxn_name = vj_name, rxn_flux_ini = vj_ini)
        v_fluxes_ini.append(v_flux_ini)
        if vj_name == fseof_target:
            vTarget_ini = vin[item]
        if vj_name == "rx0155":
            print vin[item]

    ##-------------------------------------------------------------------------
    ##2. Calculate the theorical maximum of a target specialised metabolite.
    ##-------------------------------------------------------------------------
    cobra_model.objective = fseof_target ## new objective function
    cobra_model.optimize(solver='gurobi')
    vmax_Target = cobra_model.solution.f

    ##-------------------------------------------------------------------------
    ##3. Apply FSEOF.
    ##-------------------------------------------------------------------------
    v_bio = []
    rxn = cobra_model.reactions.get_by_id(fseof_target)

    vprod_enforced_list = []
    n = float(ntimes)+1
    for k in range(1,int(ntimes)+1):
        vprod_enforced = vTarget_ini + (k/n)*(vmax_Target - vTarget_ini)
        vprod_enforced_list.append(vprod_enforced)
    i=0
    enf = {}
    enf_rx = {}
    fva_rx = {}
    #v_enforced_list = []
    v_list_rxn_enforced = []
    rr_list = []
    fva_list = []
    for item in vprod_enforced_list:
        i+=1
        rxn.lower_bound = item
        rxn.upper_bound = item
        cobra_model.objective = "ExBiomass_e"
        cobra_model.optimize(solver='gurobi')
        v_biom = cobra_model.solution.f
        v_bio.append(cobra_model.solution.f)
        vj_2 = cobra_model.solution.x_dict
        enf[i] = namedtuple("v_enforced"+str(i),["rxn_name", "rxn", "genes", "
            rxn_flux_ini", "rxn_flux_enf"])
        fva_result = cobra.flux_analysis.flux_variability_analysis(cobra_model,
                                                                   cobra_model.
                                                                       reactions[:len(
                                                                       cobra_model.
                                                                       reactions)],
                                                                   fraction_of_optimum
                                                                       =1)
        enf_rx[i] = namedtuple("v_enforced"+str(i),["rxn_name", "rxn", "genes", "
            rxn_flux_ini", "rxn_flux_enf"])
        fva_rx[i] = namedtuple("fva_min_max"+str(i),["rxn_name","vprodChax","fva_min",
            "fva_max"])
        for obj in vj_2:
            vj_2name = str(obj)
            rr = cobra_model.reactions.get_by_id(obj)
            react = rr.reaction
            gen = rr.gene_reaction_rule
```

```
                vj_au = vj_2[obj]
                for obj2 in v_fluxes_ini:
                    if vj_2name == obj2.rxn_name:
                        vj_init = obj2.rxn_flux_ini
                        if abs(vj_au) > abs(vj_init) and vj_au*vj_init >= 0:
                            ####Make a list with all the reaction numbers that had an
                                increased flux
                            if str(rr) not in rr_list:
                                rr_list.append(str(rr))

                            v_list2 = enf_rx[i](rxn_name = vj_2name, rxn = react, genes = gen,
                                 rxn_flux_ini = vj_init,
                                                rxn_flux_enf = vj_au)
                            v_list_rxn_enforced.append(v_list2)

                            rx_fva = fva_result[str(rr)]
                            rx_fva_min = rx_fva['minimum']
                            rx_fva_max = rx_fva['maximum']
                            fva_data = fva_rx[i](rxn_name = vj_2name, vprodChax = item,
                                fva_min = rx_fva_min,
                                                 fva_max = rx_fva_max)
                            fva_list.append(fva_data)

        ##Write down all the reactions that have increasing fluxes while increasing the
            FSEOF target production.
        with open(fseof_target+'FSEOF_list_general.txt', 'w') as out,\
            open(fseof_target+'FSEOF_increasing_list.txt',"w") as out2,\
            open(fseof_target+'fva_enforced_rxns.txt',"w") as out3:
            for rn in rr_list:
                rx_x_values = []
                f_min_list = []
                f_max_list = []
                for data in fva_list:
                    if rn == data.rxn_name:
                        f_min = data.fva_min
                        f_max = data.fva_max
                        f_min_list.append(f_min)
                        f_max_list.append(f_max)
                minn = '\t'.join(map(str,f_min_list))
                maxx = '\t'.join(map(str,f_max_list))
                for item in v_list_rxn_enforced:
                    if rn == item.rxn_name:
                        flux_ini = item.rxn_flux_ini
                        rxn_eq = item.rxn
                        sle = item.genes
                        rx_x_values.append(item.rxn_flux_enf)
                fluxes = '\t'.join(map(str, rx_x_values))
                out.write(str(rn)+'\t'+str(rxn_eq)+'\t'+str(sle)+'\t'+str(flux_ini)+'\t'+
                    fluxes+'\n')
                j=0
                for obj, obj2 in zip(rx_x_values, rx_x_values[1:]):
                    if abs(obj) <= abs(obj2):
                        aum = True
                        j+=1
                    elif abs(obj) > abs(obj2):
                        aum = False
                if aum == True and j == (len(rx_x_values)-1):
                    out2.write(str(rn)+'\t'+str(rxn_eq)+'\t'+str(sle)+'\t'+str(flux_ini)+'
                        \t'+fluxes+'\n')
                    out3.write(str(rn)+'\t'+'minimum'+'\t'+minn+'\n')
                    out3.write(str(rn)+'\t'+'maximum'+'\t'+maxx+'\n')
                    self.text4.insert(END, str(rn)+'\t'+str(rxn_eq)+'\t'+str(sle)+'\t'+str
                        (flux_ini)+'\t'+fluxes+'\n')
```

F.4 List of abbreviations of compounds

Table F.13: Compound abbreviations.

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00001	H2O_c	H2O	C00692	UDPNactoylLalanylDglu_c	UDP-N-acetylmuramoyl-L-alanyl-D-glutamate	C05750	tOoctoylacp_c	3-Oxooctanoyl-[acp]	M00124	t3hcddec5eACP_c	(R)-3-hydroxy-cis-dodec-5-enoyl-[acyl-carrier protein]
C00002	ATP_c	ATP	C00698	Cl_c	Cl-	C05751	traOct2enooylacp_c	trans-Oct-2-enoyl-[acp]	M00125	t3c5ddeceACP_c	trans-3-cis-5-dodecenoyl-[acyl-carrier protein]
C00003	NAD_c	NAD+	C00705	dCDP_c	dCDP	C05752	Octoylacp_c	Octanoyl-[acp]	M00126	cddec5eACP_c	cis-dodec-5-enoyl-[acyl-carrier protein] (n-C12:1)
C00004	NADH_c	NADH	C00718	Amy_c	Amylose	C05753	tOdecoylacp_c	3-Oxodecanoyl-[acp]	M00127	t3ocmrs7eACP_c	3-oxo-cis-myristol-7-eoyl-[acyl-carrier protein]
C00005	NADPH_c	NADPH	C00718	NAmy_c	Amylose_N	C05754	traDec2enooylacp_c	trans-Dec-2-enoyl-[acp]	M00128	t3hcmrs7eACP_c	(R)-3-hydroxy-cis-myristol-7-eoyl-[acyl-carrier protein]
C00006	NADP_c	NADP+	C00718	N2Amy_c	Amylose_N	C05755	Decoylacp_c	Decanoyl-[acp]	M00129	t3c7mrseACP_c	trans-3-cis-7-myristoleoyl-[acyl-carrier protein]
C00007	O2_c	Oxygen	C00719	Bet_c	Betaine	C05756	tOdodecoylacp_c	3-Oxododecanoyl-[acp]	M00130	tdeACP_c	cis-tetradec-7-enoyl-[acyl-carrier protein] (n-C14:1)
C00008	ADP_c	ADP	C00721	Dex_c	Dextrin	C05757	R3Hxydecoylacp_c	(R)-3-Hydroxydodecanoyl-[acp]	M00131	t3ocpalm9eACP_c	3-oxo-cis-palm-9-eoyl-[acyl-carrier protein]
C00009	pi_c	Orthophosphate	C00721_	Dex__c	Dextrin	C05758	traDod2enooylacp_c	trans-Dodec-2-enoyl-[acp]	M00132	t3hcpalm9eACP_c	(R)-3-hydroxy-cis-palm-9-eoyl-[acyl-carrier protein]
C00010	CoA_c	CoA	C00721__	Dex___c	Dextrin	C05759	tOtetdecoylacp_c	3-Oxotetradecanoyl-[acp]	M00133	t3c9palmeACP_c	trans-3-cis-9-palmitoleoyl-[acyl-carrier protein]
C00011	CO2_c	CO2	C00760	Cel_c	Cellulose	C05760	traTde2enooylacp_c	trans-Tetradec-2-enoyl-[acp]	M00134	hdeACP_c	cis-hexadec-9-enoyl-[acyl-carrier protein] (n-C16:1)
C00013	ppi_c	Diphosphate	C00760	Cel__c	Cellulose	C05761	Tdecoylacp_c	Tetradecanoyl-[acp]	M00135	t3ocvac11eACP_c	3-oxo-cis-vacc-11-enoyl-[acyl-carrier protein]
C00014	NH3_c	Ammonia	C00785	Uro_c	Urocanate	C05762	tOhexdecoylacp_c	3-Oxohexadecanoyl-[acp]	M00136	t3hcvac11eACP_c	(R)-3-hydroxy-cis-vacc-11-enoyl-[acyl-carrier protein]
C00015	UDP_c	UDP	C00794	DSor_c	D-Sorbitol	C05763	traHex2enooylacp_c	trans-Hexadec-2-enoyl-[acp]	M00137	t3c11vaceACP_c	trans-3-cis-11-vacceoyl-[acyl-carrier protein]
C00016	FAD_c	FAD	C00800	LGul_c	L-Gulonate	C05764	Hexdecoylacp_c	Hexadecanoyl-[acp]	M00138	octeACP_c	cis-octadec-11-enoyl-[acyl-carrier protein] (n-C18:1)
C00019	SAdeLmet_c	S-Adenosyl-L-methionine	C00818	DGluca_c	D-Glucarate	C05766	UropI_c	Uroporphyrinogen I	M00139	t3optnACP_c	3-Oxopentanoyl-[acyl-carrier protein]
C00020	AMP_c	AMP	C00822	Dop_c	Dopaquinone	C05768	CoprI_c	Coproporphyrinogen I	M00140	t3hptnACP_c	(R)-3-Hydroxypentanoyl-[acyl-carrier protein]
C00021	SAdeLhom_c	S-Adenosyl-L-homocysteine	C00828	Men_c	Menaquinone	C05772	Prec3A_c	Precorrin 3A	M00141	tptn2eACP_c	trans-pent-2-enoyl-[acyl-carrier protein]
C00022	Pyr_c	Pyruvate	C00831	Ptteine_c	Pantetheine	C05773	Coby_c	Cobyrinate	M00142	ptnACP_c	Pentanoyl-ACP (n-C5:0ACP)
C00024	AceCoA_c	Acetyl-CoA	C00842	dTDglu_c	dTDP-glucose	C05774	Cob_c	Cobinamide	M00143	t3ohepACP_c	3-Oxoheptanoyl-[acyl-carrier protein]
C00025	LGlu_c	L-Glutamate	C00846	tOadi_c	3-Oxoadipate	C05775	aRib_c	alpha-Ribazole	M00144	t3hhepACP_c	(R)-3-Hydroxyheptanoyl-[acyl-carrier protein]
C00026	dOglu_c	2-Oxoglutarate	C00856	DNAcyto_c	DNA cytosine	C05778	Sir_c	Sirohydrochlorin	M00145	thep2eACP_c	trans-hep-2-enoyl-[acyl-carrier protein]
C00027	H202_c	Hydrogen peroxide	C00857	DeaNAD_c	Deamino-NAD+	C05809	Octnyl4hxybenzoate_c	3-Octaprenyl-4-hydroxybenzoate	M00146	hepACP_c	Heptanoyl-ACP (n-C7:0ACP)
C00028	Acc_c	Acceptor	C00860	LHisnol_c	L-Histidinol	C05810	Octylphenol_c	2-Octaprenylphenol	M00147	t3ononACP_c	3-Oxononanoyl-[acyl-carrier protein]
C00029	UDPglu_c	UDP-glucose	C00861	LRham_c	L-Rhamnulose	C05811	Octyl6hxyphenol_c	2-Octaprenyl-6-hydroxyphenol	M00148	t3hnonACP_c	(R)-3-Hydroxynonanoyl-[acyl-carrier protein]
C00030	Reduacce_c	Reduced acceptor	C00864	Pan_c	Pantothenate	C05812	Octyl6mxyphenol_c	2-Octaprenyl-6-methoxyphenol	M00149	tnon2eACP_c	trans-non-2-enoyl-[acyl-carrier protein]
C00031	DGlu_c	D-Glucose	C00876	CoenF420_c	Coenzyme F420	C05813	Octyl6mxy14benzoquinone_c	2-Octaprenyl-6-methoxy-1,4-benzoquinone	M00150	nonACP_c	Nonanoyl-ACP (n-C9:0ACP)
C00032	Heme_c	Heme	C00877	CrooylCoA_c	Crotonoyl-CoA	C05814	Octyl3myl6mxy14benzoquinone_c	2-Octaprenyl-3-methyl-6-methoxy-1,4-benzoquinone	M00151	t3oundcaACP_c	3-Oxoundecanoyl-[acyl-carrier protein]
C00033	Ace_c	Acetate	C00881	Dxc_c	Deoxycytidine	C05815	Octyl3myl5hxy6mxy14benzoquinone_c	2-Octaprenyl-3-methyl-5-hydroxy-6-methoxy-1,4-benzoquinone	M00152	t3hundcaACP_c	(R)-3-Hydroxyundecanoyl-[acyl-carrier protein]

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00034	Mn2_c	Manganese	C00882	DppCoA_c	Dephospho-CoA	C05817	RRHxy2succylcychex24dene1carlate_c	(1R,6R)-6-Hydroxy-2-succinylcyclohexa-2,4-diene-1-carboxylate	M00153	tundca2eACP_c	trans-undec-2-enoyl-[acyl-carrier protein]
C00035	GDP_c	GDP	C00885	iCho_c	Isochorismate	C05818	Demethylmen_c	2-Demethylmenaquinone	M00154	undcaACP_c	Undecanoyl-ACP (n-C11:0ACP)
C00036	Oxa_c	Oxaloacetate	C00886	LAlanyltRN_c	L-Alanyl-tRNA	C05819	Mennol_c	Menaquinol	M00155	t3otridcaACP_c	3-Oxotridecanoyl-[acyl-carrier protein]
C00037	Gly_c	Glycine	C00894	PropoylCoA_c	Propenoyl-CoA	C05824	SSulLcys_c	S-Sulfo-L-cysteine	M00156	t3htridcaACP_c	(R)-3-Hydroxytridecanoyl-[acyl-carrier protein]
C00038	Zn2_c	Zinc cation	C00900	dAce_c	2-Acetolactate	C05840	Imi_c	Iminoaspartate	M00157	ttridca2eACP_c	trans-tridec-2-enoyl-[acyl-carrier protein]
C00039	DNA_c	DNA	C00921	Dihte_c	Dihydropteroate	C05841	NicDrib_c	Nicotinate D-ribonucleoside	M00158	tridcaACP_c	Tridecanoyl-ACP (n-C13:0ACP)
C00041	LAla_c	L-Alanine	C00931	Por_c	Porphobilinogen	C05852	dHxynyl_c	2-Hydroxyphenylacetate	M00159	t3optdcaACP_c	3-Oxopentadecanoyl-[acyl-carrier protein]
C00042	Succ_c	Succinate	C00942	t5CyclGMP_c	3',5'-Cyclic GMP	C05892	UDPNactoylLalanylgamDgluLlys_c	UDP-N-acetylmuramoyl-L-alanyl-gamma-D-glutamyl-L-lysine	M00160	t3hptdcaACP_c	(R)-3-Hydroxypentadecanoyl-[acyl-carrier protein]
C00043	UDPNaceaDglu_c	UDP-N-acetyl-alpha-D-glucosamine	C00944	tDhq_c	3-Dehydroquinate	C05893	UndnyldipNactoylNaceLalanylgamDgluLlysDalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-gamma-D-glutamyl-L-lysyl-D-alanyl-D-alanine	M00161	tptdca2eACP_c	trans-pentadec-2-enoyl-[acyl-carrier protein]
C00044	GTP_c	GTP	C00957	Mer_c	Mercaptopyruvate	C05894	UndnyldipNactoylNaceLalanylDisoglunylLlysDalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-D-isoglutaminyl-L-lysyl-D-alanyl-D-alanine	M00162	ptdcaACP_c	Pentadecanoyl-ACP (n-C15:0ACP)
C00046	RNA_c	RNA	C00966	dDhpte_c	2-Dehydropantoate	C05895	UndnyldipNactoylNaceLalanylDisoglunylLlysglycyl5DalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-D-isoglutaminyl-L-lysyl-(glycyl)5-D-alanyl-D-alanine	M00163	t3ohepdcaACP_c	3-Oxoheptadecanoyl-[acyl-carrier protein]
C00047	LLys_c	L-Lysine	C00979	OAceLser_c	O-Acetyl-L-serine	C05897	UndnyldipNactoylLalanylDglumes26diaoylDalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-L-alanyl-D-glutamyl-meso-2,6-diaminopimeloyl-D-alanyl-D-alanine	M00164	t3hhepdcaACP_c	(R)-3-Hydroxyheptadecanoyl-[acyl-carrier protein]
C00048	Glyo_c	Glyoxylate	C00984	aDGal_c	alpha-D-Galactose	C05898	UndnyldipNactoylNaceLalanylDglumes26diaoylDalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-D-glutamyl-meso-2,6-diaminopimeloyl-D-alanyl-D-alanine	M00165	thepdca2eACP_c	trans-heptadec-2-enoyl-[acyl-carrier protein]
C00049	LAsp_c	L-Aspartate	C00986	u3Dia_c	1,3-Diaminopropane	C05921	Bionyl5AMP_c	Biotinyl-5'-AMP	M00166	hepdcaACP_c	Heptadecanoyl-ACP (n-C17:0ACP)
C00051	Glu_c	Glutathione	C00988	dPgl_c	2-Phosphoglycolate	C05922	Formnucltrip_c	Formamidopyrimidine nucleoside triphosphate	M00167	ciC112eACP_c	cis-isoundec-2-enoyl-[acyl-carrier protein]
C00052	UDPaDgal_c	UDP-alpha-D-galactose	C00993	DAlanylDala_c	D-Alanyl-D-alanine	C05923	d5Diamnucltrip_c	2,5-Diaminopyrimidine nucleoside triphosphate	M00168	iC111ACP_c	Isoundecenoyl-ACP (iso-C11:1ACP)
C00053	tPadesulf_c	3'-Phosphoadenylyl sulfate	C01005	OPLser_c	O-Phospho-L-serine	C05933	NoHxy_c	N(omega)-Hydroxyarginine	M00169	t3ociC13ACP_c	3-Oxo-cis-isotridecenoyl-[acyl-carrier protein]
C00054	Aden35bp_c	Adenosine 3',5'-bisphosphate	C01010	Urea1car_c	Urea-1-carboxylate	C05936	N4Aceaminal_c	N4-Acetylaminobutanal	M00170	t3hciC13ACP_c	(R)-3-Hydroxy-cis-isotridecenoyl-[acyl-carrier protein]
C00055	CMP_c	CMP	C01019	sDxLgal_c	6-Deoxy-L-galactose	C05938	L4Hxygsald_c	L-4-Hydroxyglutamate semialdehyde	M00171	tciC13eACP_c	trans-3-cis-isotridecenoyl-[acyl-carrier protein]
C00058	For_c	Formate	C01024	Hxybil_c	Hydroxymethylbilane	C05946	qR4Hxy2ogl_c	(4R)-4-Hydroxy-2-oxoglutarate	M00172	iC131ACP_c	Isotridecenoyl-ACP (iso-C13:1ACP)
C00059	Sulf_c	Sulfate	C01033	dmbcoa_c	2-Methylbutanoyl-CoA	C05947	Lery4Hxy_c	L-erythro-4-Hydroxyglutamate	M00173	t3ociC15ACP_c	3-Oxo-cis-isopentadecenoyl-[acyl-carrier protein]
C00061	FMN_c	FMN	C01036	qMal_c	4-Maleylacetoacetate	C05980	Card_c	Cardiolipin	M00174	t3hciC15ACP_c	(R)-3-Hydroxy-cis-isopentadecenoyl-[acyl-carrier protein]

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00062	LArg_c	L-Arginine	C01037	ste8Diate_c	7,8-Diaminononanoate	C05983	Pronyladenyl_c	Propionyladenylate	M00175	tciC15eACP_c	trans-cis-isopentadecenoyl-[acyl-carrier protein]
C00063	CTP_c	CTP	C01040	LGul14lac_c	L-Gulono-1,4-lactone	C05984	dHxybad_c	2-Hydroxybutanoic acid	M00176	iC151ACP_c	Isopentadecenoyl-ACP (iso-C15:1ACP)
C00064	LGlut_c	L-Glutamine	C01050	UDPNace_c	UDP-N-acetylmuramate	C05993	Acetaden_c	Acetyl adenylate	M00177	t3ociC17ACP_c	3-Oxo-cis-isoheptadecenoyl-[acyl-carrier protein]
C00065	LSer_c	L-Serine	C01051	UropIII_c	Uroporphyrinogen III	C05998	tHxyisovalCoA_c	3-Hydroxyisovaleryl-CoA	M00178	t3hciC17ACP_c	(R)-3-Hydroxy-cis-isoheptadecenoyl-[acyl-carrier protein]
C00067	Forlde_c	Formaldehyde	C01061	qFum_c	4-Fumarylacetoacetate	C06000	S3HxyisobutCoA_c	(S)-3-Hydroxyisobutyryl-CoA	M00179	tciC17eACP_c	trans-iso-iso-heptadecenoyl-[acyl-carrier protein]
C00068	Thiadp_c	Thiamin diphosphate	C01063	sCaroylCoA_c	6-Carboxyhexanoyl-CoA	C06001	S3Hxyisobut_c	(S)-3-Hydroxyisobutyrate	M00180	iC171w9ACP_c	Isoheptadecenoyl-ACP (iso-C17:1w9ACP)
C00070	Cu2_c	Copper	C01077	OAceLhom_c	O-Acetyl-L-homoserine	C06002	SMylmalosald_c	(S)-Methylmalonate semi-aldehyde	M00181	caiC112e_c	cis-ante-isoundec-2-enoyl-[acyl-carrier protein]
C00072	Asc_c	Ascorbate	C01079	ProtIX_c	Protoporphyrinogen IX	C06006	S2Ace2hydte_c	(S)-2-Aceto-2-hydroxybutanoate	M00182	aiC111ACP_c	Ante-isoundecenoyl-ACP (ante-iso-C11:1ACP)
C00073	LMet_c	L-Methionine	C01081	Thiamono_c	Thiamin monophosphate	C06007	R23Dih3mylpente_c	(R)-2,3-Dihydroxy-3-methylpentanoate	M00183	t3ocaiC13ACP_c	3-Oxo-cis-ante-isotridecanoyl-[acyl-carrier protein]
C00074	Pennolpyr_c	Phosphoenolpyruvate	C01083	aaTre_c	alpha,alpha-Trehalose	C06010	S2Ace_c	(S)-2-Acetolactate	M00184	t3hcaiC13ACP_c	(R)-3-Hydroxy-cis-ante-isotridecenoyl-[acyl-carrier protein]
C00075	UTP_c	UTP	C01089	R3Hxyte_c	(R)-3-Hydroxybutanoate	C06032	Dery3Mylmal_c	D-erythro-3-Methylmalate	M00185	tcaiC13eACP_c	trans-cis-ante-isotridecenoyl-[acyl-carrier protein]
C00076	Ca2_c	Calcium cation	C01092	oAmi7onote_c	8-Amino-7-oxononanoate	C06148	d5Dia65tri34tri2-opeami4opy_c	2,5-Diamino-6-(5'-triphosphoryl-3',4'-trihydroxy-2'-oxopentyl)-amino-4-oxopyrimidine	M00186	aiC131ACP_c	Ante-isotridecenoyl-ACP (ante-iso-C13:1ACP)
C00077	LOrn_c	L-Ornithine	C01094	DFru1p_c	D-Fructose 1-phosphate	C06156	aDGluc1p_c	alpha-D-Glucosamine 1-phosphate	M00187	t3ocaiC15ACP_c	3-Oxo-cis-ante-isopentadecenoyl-[acyl-carrier protein]
C00078	LTry_c	L-Tryptophan	C01097	DTaga6p_c	D-Tagatose 6-phosphate	C06157	DihresisuccSglry_c	[Dihydrolipoyllysine-residue succinyltransferase] S-glutaryldihydrolipoyllysine	M00188	t3hcaiC15ACP_c	(R)-3-Hydroxy-cis-ante-isopentadecenoyl-[acyl-carrier protein]
C00079	LPhenylala_c	L-Phenylalanine	C01099	LFucu1p_c	L-Fuculose 1-phosphate	C06187	Arbu6p_c	Arbutin 6-phosphate	M00189	tcaiC15eACP_c	trans-cis-ante-isopentadecenoyl-[acyl-carrier protein]
C00080	H_c	H+	C01100	LHisp_c	L-Histidinol phosphate	C06188	Sal6p_c	Salicin 6-phosphate	M00190	aiC151ACP_c	Ante-isopentadecenoyl-ACP (ante-iso-C15:1ACP)
C00081	ITP_c	ITP	C01101	LRib5p_c	L-Ribulose 5-phosphate	C06231	Ect_c	Ectoine	M00191	t3ocaiC17ACP_c	3-Oxo-cis-ante-isoheptadecenoyl-[acyl-carrier protein]
C00082	LTyr_c	L-Tyrosine	C01102	OPLhom_c	O-Phospho-L-homoserine	C06232	Molbd_c	Molybdate	M00192	t3hcaiC17ACP_c	(R)-3-Hydroxy-cis-ante-isoheptadecenoyl-[acyl-carrier protein]
C00083	MalnylCoA_c	Malonyl-CoA	C01103	Orot5p_c	Orotidine 5'-phosphate	C06249	Apoca_c	Apo-[carboxylase]	M00193	tcaiC17eACP_c	trans-cis-ante-isoheptadecenoyl-[acyl-carrier protein]
C00084	Acelde_c	Acetaldehyde	C01107	R5Pme_c	(R)-5-Phosphomevalonate	C06250	Holca_c	Holo-[carboxylase]	M00194	aiC171w9ACP_c	Ante-isoheptadecenoyl-ACP (ante-iso-C17:1w9ACP)
C00085	DFru6p_c	D-Fructose 6-phosphate	C01118	OSucnylLhom_c	O-Succinyl-L-homoserine	C06319	Prec6Y_c	Precorrin 6Y	M00195	M00195_c	cis-undec-2-enoyl-[acyl-carrier protein]
C00086	Urea_c	Urea	C01131	LRha1p_c	L-Rhamnulose 1-phosphate	C06320	Prec6X_c	Precorrin 6X	M00196	C111ACP_c	Undecenoyl-ACP (n-C11:1ACP)
C00088	Nit_c	Nitrite	C01134	Pant4p_c	Pantetheine 4'-phosphate	C06399	Hge_c	Hydrogenobyrinate	M00197	t3octridceACP_c	3-Oxo-cis-tridecenoyl-[acyl-carrier protein]
C00089	Suc_c	Sucrose	C01142	tS36Diate_c	(3S)-3,6-Diaminohexanoate	C06406	Prec3B_c	Precorrin 3B	M00198	t3hctridceACP_c	(R)-3-Hydroxy-cis-tridecenoyl-[acyl-carrier protein]
C00090	Cat_c	Catechol	C01143	R5Dpm_c	(R)-5-Diphosphomevalonate	C06407	Prec4_c	Precorrin 4	M00199	tctidceACP_c	trans-cis-tridecenoyl-[acyl-carrier protein]
C00091	SucnylCoA_c	Succinyl-CoA	C01144	S3HxyoylCoA_c	(S)-3-Hydroxybutanoyl-CoA	C06408	Prec8X_c	Precorrin 8X	M00200	C131ACP_c	Tridecenoyl-ACP (n-C13:1ACP)
C00092	DGlu6p_c	D-Glucose 6-phosphate	C01146	dHxy3oprte_c	2-Hydroxy-3-oxopropanoate	C06416	Prec5_c	Precorrin 5	M00201	t3coptdceACP_c	3-Oxo-cis-pentadecenoyl-[acyl-carrier protein]

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00093	snGlyc3p_c	sn-Glycerol 3-phosphate	C01157	Hxy_c	Hydroxyproline	C06424	C140_c	tetradecanoate (C14:0)	M00202	t3hcptdceACP_c	(R)-3-Hydroxy-cis-pentadecenoyl-[acyl-carrier protein]
C00094	Sul_c	Sulfite	C01163	tCarcismuc_c	3-Carboxy-cis,cis-muconate	C06442	NgAcedia_c	N(gamma)-Acetyldiaminobutyrate	M00203	tctptdceACP_c	trans-cis-pentadecenoyl-[acyl-carrier protein]
C00095	DFru_c	D-Fructose	C01165	LGlu5sald_c	L-Glutamate 5-semialdehyde	C06503	Hgeacdiad_c	Hydrogenobyrinate a,c diamide	M00204	C151ACP_c	Pentadecenoyl-ACP (n-C15:1ACP)
C00096	GDPman_c	GDP-mannose	C01168	Psuri5p_c	Pseudouridine 5'-phosphate	C06504	CobIacdiade_c	Cob(II)yrinate a,c diamide	M00205	t3ochepdceACP_c	3-Oxo-cis-heptadecenoyl-[acyl-carrier protein]
C00097	LCys_c	L-Cysteine	C01172	bDGluc6p_c	beta-D-Glucose 6-phosphate	C06505	Cobacdiad_c	Cob(I)yrinate a,c diamide	M00206	t3hchepdceACP_c	(R)-3-Hydroxy-cis-heptadecenoyl-[acyl-carrier protein]
C00099	bAla_c	beta-Alanine	C01177	Inos1p_c	Inositol 1-phosphate	C06506	Adencobyacdiad_c	Adenosyl cobyrinate a,c diamide	M00207	tchepdceACP_c	trans-cis-heptadecenoyl-[acyl-carrier protein]
C00100	ProoylCoA_c	Propanoyl-CoA	C01179	t4Hxynyl_c	3-(4-Hydroxyphenyl)pyruvate	C06507	Adencobyhexa_c	Adenosyl cobyrinate hexaamide	M00208	C171ACP_c	Heptadecenoyl-ACP (n-C17:1ACP)
C00101	Thy_c	Tetrahydrofolate	C01185	NicotDribon_c	Nicotinate D-ribonucleotide	C06508	Adencobi_c	Adenosyl cobinamide	M00209	iC14p_c	Isotetradecanoyl-phosphate (iso-C14:0)
C00103	DGlu1p_c	D-Glucose 1-phosphate	C01186	tS5S35Diate_c	(3S,5S)-3,5-Diaminohexanoate	C06509	Adencobip_c	Adenosyl cobinamide phosphate	M00210	uiC14g3p_c	1-isotetradecanoyl-sn-glycerol 3-phosphate
C00104	IDP_c	IDP	C01194	uPtiDmyoino_c	1-Phosphatidyl-D-myo-inositol	C06510	AdenGDPcob_c	Adenosine-GDP-cobinamide	M00211	iC14_c	Isotetradecanoate (iso-C14:0)
C00105	UMP_c	UMP	C01203	Oleoylacp_c	Oleoyl-[acyl-carrier protein]	C06892	dDx5ketoDglucad_c	2-Deoxy-5-keto-D-gluconic acid	M00212	pai140_c	1,2-diisotetradecanoyl-sn-glycerol 3-phosphate
C00106	Ura_c	Uracil	C01209	Malnylacp_c	Malonyl-[acyl-carrier protein]	C06893	dDx5ketoDglucad-6p_c	2-Deoxy-5-keto-D-gluconic acid 6-phosphate	M00213	iC16p_c	Isohexadecanoyl-phosphate (n-C16:0)
C00108	Ant_c	Anthranilate	C01212	UDPNactoylLala_c	UDP-N-acetylmuramoyl-L-alanine	C07086	Phenacetad_c	Phenylacetic acid	M00214	uiC16g3p_c	1-isohexadecanoyl-sn-glycerol 3-phosphate
C00109	dObutte_c	2-Oxobutanoate	C01213	RMylmalnylCoA_c	(R)-Methylmalonyl-CoA	C08362	hdcea_c	Hexadecenoate (n-C16:1)	M00215	iC16_c	Isohexadecanoate (n-C16:0)
C00111	Glycp_c	Glycerone phosphate	C01217	c678Thy_c	5,6,7,8-Tetrahydromethanopterin	C09332	THFLglu_c	THF-L-glutamate	M00216	pai160_c	1,2-diisohexadecanoyl-sn-glycerol 3-phosphate
C00112	CDP_c	CDP	C01222	GDP4deh6deoD-man_c	GDP-4-dehydro-6-deoxy-D-mannose	C11355	qAmi4deo_c	4-Amino-4-deoxychorismate	M00217	iC18p_c	Isooctadecanoyl-phosphate (n-C18:0)
C00114	Chol_c	Choline	C01228	Guan35bis_c	Guanosine 3',5'-bis(diphosphate)	C11434	dCMylDeryt4p_c	2-C-Methyl-D-erythritol 4-phosphate	M00218	uiC18g3p_c	1-isooctadecanoyl-sn-glycerol 3-phosphate
C00116	Glyc_c	Glycerol	C01230	alltHexyldp_c	all-trans-Hexaprenyl diphosphate	C11435	qCyt5diph2CmylD-eryt_c	4-(Cytidine 5'-diphospho)-2-C-methyl-D-erythritol	M00219	iC18_c	Isooctadecanoate (n-C18:0)
C00117	DRibo5p_c	D-Ribose 5-phosphate	C01235	aDGal131Dmyo-ino_c	alpha-D-Galactosyl-(1-¿3)-1D-myo-inositol	C11436	dP4cyt5diph2Cmyl-Deryt_c	2-Phospho-4-(cytidine 5'-diphospho)-2-C-methyl-D-erythritol	M00220	pai180_c	1,2-diisooctadecanoyl-sn-glycerol 3-phosphate
C00118	DGlyc3p_c	D-Glyceraldehyde 3-phosphate	C01236	DGluc15lact6p_c	D-Glucono-1,5-lactone 6-phosphate	C11437	uDxDxylu5p_c	1-Deoxy-D-xylulose 5-phosphate	M00221	iC15p_c	Isopentadecanoyl-phosphate (iso-C15:0)
C00119	cPaDribo1dp_c	5-Phospho-alpha-D-ribose 1-diphosphate	C01242	SAmidihoyl_c	S-Aminomethyldihydrolipoyl-protein	C11453	dCMylDeryt24-cycl_c	2-C-Methyl-D-erythritol 2,4-cyclodiphosphate	M00222	uiC15g3p_c	1-isopentadecanoyl-sn-glycerol 3-phosphate
C00120	Bio_c	Biotin	C01250	NAcetLglut5sald_c	N-Acetyl-L-glutamate 5-semialdehyde	C11538	Cobsir_c	Cobalt-sirohydrochlorin	M00223	iC15_c	Isopentadecanoate (iso-C15:0)
C00121	DRib_c	D-Ribose	C01267	tImi4yl2oprop_c	3-(Imidazol-4-yl)-2-oxopropyl phosphate	C11539	Cobaprec3_c	Cobalt-precorrin 3	M00224	pai150_c	1,2-diisopentadecanoyl-sn-glycerol 3-phosphate
C00122	Fum_c	Fumarate	C01268	cAmi65pho_c	5-Amino-6-(5'-phosphoribosylamino)uracil	C11540	Cobaprec4_c	Cobalt-precorrin 4	M00225	iC17p_c	Isoheptadecanoyl-phosphate (iso-C17:0)
C00123	LLeu_c	L-Leucine	C01269	cO1Carnyl3pho_c	5-O-(1-Carboxyvinyl)-3-phosphoshikimate	C11542	Cobaprec6_c	Cobalt-precorrin 6	M00226	uiC17g3p_c	1-isoheptadecanoyl-sn-glycerol 3-phosphate
C00124	DGal_c	D-Galactose	C01278	dCar25dih5ofu2-ace_c	2-Carboxy-2,5-dihydro-5-oxofuran-2-acetate	C11543	Cobadihyprec6_c	Cobalt-dihydro-precorrin 6	M00227	iC17_c	Isoheptadecanoate (iso-C17:0)
C00125	Ferrc_c	Ferricytochrome c	C01300	dAmi4hyd6hyd78-dih_c	2-Amino-4-hydroxy-6-hydroxymethyl-7,8-dihydropteridine	C11545	Cobaprec8_c	Cobalt-precorrin 8	M00228	pai170_c	1,2-diisoheptadecanoyl-sn-glycerol 3-phosphate
C00126	Ferroc_c	Ferrocytochrome c	C01302	u2Carb1deoxDribu-5p_c	1-(2-Carboxyphenylamino)-1-deoxy-D-ribulose 5-phosphate	C11811	uHxy2myl2bute4-dp_c	1-Hydroxy-2-methyl-2-butenyl 4-diphosphate	M00229	aiC15p_c	Ante-isopentadecanoyl-phosphate (ante-iso-C15:0)
C00129	Isopentdp_c	Isopentenyl diphosphate	C01304	d5Dia65phoDrbs43-Hone_c	2,5-Diamino-6-(5-phospho-D-ribosylamino)pyrimidin-4(3H)-one	C11821	cHxyisoura_c	5-Hydroxyisourate	M00230	uaiC15g3p_c	1-ante-isopentadecanoyl-sn-glycerol 3-phosphate

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00130	IMP_c	IMP	C01330	Na_c	Sodium cation	C11827	Gl14Mur2Acoyl-LAlagDGluA2pm-DAlaDAladipnol_c	[GlcNAc-(1-¿4)-Mur2Ac(oyl-L-Ala-g-D-Glu-A2pm-D-Ala-D-Ala)]n-diphosphoundecaprenol	M00231	aiC15_c	Ante-isopentadecanoate (ante-iso-C15:0)
C00131	dATP_c	dATP	C01346	dUDP_c	dUDP	C11827	Gl14Mur2Acoyl-LAlagDGluA2pm-DAlaDAladipnol__c	[GlcNAc-(1-¿4)-Mur2Ac(oyl-L-Ala-g-D-Glu-A2pm-D-Ala-D-Ala)]n-diphosphoundecaprenol	M00232	paai150_c	1,2-diante-isopentadecanoyl-sn-glycerol 3-phosphate
C00132	Metnol_c	Methanol	C01352	FADH2_c	FADH2	C11907	dTD4o6deoDglu_c	dTDP-4-oxo-6-deoxy-D-glucose	M00233	aiC17p_c	Ante-isoheptadecanoyl-phosphate (ante-iso-C17:0)
C00133	DAla_c	D-Alanine	C01417	Cya_c	Cyanate	C12106	AmiD_c	AminoDAHP	M00234	uaiC17g3p_c	1-ante-isoheptadecanoyl-sn-glycerol 3-phosphate
C00134	Put_c	Putrescine	C01419	CysGly_c	Cys-Gly	C12107	tAmi5hydbzte_c	3-Amino-5-hydroxybenzoate	M00235	aiC17_c	Ante-isoheptadecanoate (ante-iso-C17:0)
C00135	LHis_c	L-Histidine	C01530	C180_c	octadecanoate (n-C18:0)	C12108	cAmi5deo3deh_c	5-Amino-5-deoxy-3-dehydroshikimate	M00236	paai170_c	1,2-diante-isoheptadecanoyl-sn-glycerol 3-phosphate
C00136	ButoylCoA_c	Butanoyl-CoA	C01563	Car_c	Carbamate	C12109	Ami_c	AminoDHQ	M00237	hdceap_c	Hexadecenoyl-phosphate (n-C16:1)
C00137	myoIno_c	myo-Inositol	C01613	Sta_c	Stachyose	C12147	LThreO3p_c	L-Threonine O-3-phosphate	M00238	uhdec9eg3p_c	1-hexadec-9-enoyl-sn-glycerol 3-phosphate
C00138	Reduferr_c	Reduced ferredoxin	C01635	tRNAAla_c	tRNA(Ala)	C12210	UDP3ket_c	UDP-3-ketoglucose	M00239	pa161_c	1,2-dihexadec-9-enoyl-sn-glycerol 3-phosphate
C00139	Oxidferr_c	Oxidized ferredoxin	C01641	tRNAGlu_c	tRNA(Glu)	C12211	UDPkan_c	UDP-kanosamine	M00240	ptdcap_c	Pentadecanoyl-phosphate (n-C15:0)
C00140	NAceDglu_c	N-Acetyl-D-glucosamine	C01642	tRNAGly_c	tRNA(Gly)	C12212	Kan_c	Kanosamine	M00241	uptdecg3p_c	1-pentadecanoyl-sn-glycerol 3-phosphate
C00141	tMyl2obutad_c	3-Methyl-2-oxobutanoic acid	C01672	Cad_c	Cadaverine	C12213	Kano6p_c	Kanosamine 6-phosphate	M00242	pa150_c	1,2-dipentadecanoyl-sn-glycerol 3-phosphate
C00143	c10Mylenehfol_c	5,10-Methylenetetrahydrofolate	C01674	Chi_c	Chitobiose	C12214	Amif6p_c	Aminofructose 6-phosphate	M00243	hepdcap_c	heptadecanoyl-phosphate (n-C17:0)
C00144	GMP_c	GMP	C01720	LFuc_c	L-Fuconate	C12215	Imin4p_c	Iminoerythrose 4-phosphate	M00244	uhepdecg3p_c	1-heptadecanoyl-sn-glycerol 3-phosphate
C00147	Ade_c	Adenine	C01721	LFucu_c	L-Fuculose	C12248	cHxy2o4ure25dih1-Himi5car_c	5-Hydroxy-2-oxo-4-ureido-2,5-dihydro-1H-imidazole-5-carboxylate	M00245	hepdca_c	heptadecanoate (n-C17:0)
C00148	LPro_c	L-Proline	C01755	Thioc_c	Thiocyanate	C14144	cCar2penoylCoA_c	5-Carboxy-2-pentenoyl-CoA	M00246	pa170_c	1,2-diheptadecanoyl-sn-glycerol 3-phosphate
C00149	SMal_c	(S)-Malate	C01762	Xant_c	Xanthosine	C14145	tS3HxyCoA_c	(3S)-3-Hydroxyadipyl-CoA	M00247	u2dgri140_c	1,2-Diacyl-sn-glycerol (di-isotetradecanoyl, iso-C14:0)
C00152	LAspa_c	L-Asparagine	C01801	Dxr_c	Deoxyribose	C14463	R3Hxy3myl2ope-te_c	(R)-3-Hydroxy-3-methyl-2-oxopentanoate	M00248	tagi140_c	Triacylglycerol (triisotetradecanoyl, iso-C14:0)
C00153	Nic_c	Nicotinamide	C01832	LauoylCoA_c	Lauroyl-CoA	C14610	S5O25dih2ace_c	(S)-5-Oxo-2,5-dihydrofuran-2-acetate	M00249	u2dgri160_c	1,2-Diacyl-sn-glycerol (di-isohexadecanoyl, iso-C16:0)
C00154	PaloylCoA_c	Palmitoyl-CoA	C01847	ReduFMN_c	Reduced FMN	C14818	Fe2_c	Fe2+	M00250	tagi160_c	Triacylglycerol (triisohexadecanoyl, iso-C16:0)
C00155	LHom_c	L-Homocysteine	C01879	Pidoad_c	Pidolic acid	C14818	Fe2_c	Fe2+	M00251	u2dgri180_c	1,2-Diacyl-sn-glycerol (di-isooctadecanoyl, iso-C18:0)
C00156	qHxybzte_c	4-Hydroxybenzoate	C01909	Det_c	Dethiobiotin	C14819	Fe3_c	Fe3+	M00252	tagi180_c	Triacylglycerol (triisooctadecanoyl, iso-C18:0)
C00157	Ptid_c	Phosphatidylcholine	C01929	LHisnal_c	L-Histidinal	C15494	Decubi_c	Decylubiquinone	M00253	cdpdiC14g_c	CDP-1,2-diisotetradecanoylglycerol
C00158	Cit_c	Citrate	C01944	OctoylCoA_c	Octanoyl-CoA	C15495	Decnol_c	Decylubiquinol	M00254	cdpdiC16g_c	CDP-1,2-diisohexadecanoylglycerol
C00159	DMan_c	D-Mannose	C01990	tOxa_c	3-Oxalomalate	C15547	Dhxy2naphoyl-CoA_c	1,4-Dihydroxy-2-naphthoyl-CoA	M00255	cdpdiC18g_c	CDP-1,2-diisooctadecanoylglycerol
C00160	Glyco_c	Glycolate	C02051	Lipoyl_c	Lipoylprotein	C15556	L34Dihy2one4p_c	L-3,4-Dihydroxybutan-2-one 4-phosphate	M00256	u2dgri150_c	1,2-Diacyl-sn-glycerol (di-isopentadecanoyl, iso-C15:0)
C00163	Prote_c	Propanoate	C02084	Tthi_c	Tetrathionate	C15602	Qui_c	Quinone	M00257	tagi150_c	Triacylglycerol (triisopentadecanoyl, iso-C15:0)
C00164	Acet_c	Acetoacetate	C02170	Mylmal_c	Methylmalonate	C15603	Hqui_c	Hydroquinone	M00258	cdpdiC15g_c	CDP-1,2-diisopentadecanoylglycerol
C00166	Phenylpyr_c	Phenylpyruvate	C02191	Pro_c	Protoporphyrin	C15606	u2Dih5mythi1en3-one_c	1,2-Dihydroxy-5-(methylthio)pent-1-en-3-one	M00259	u2dgri170_c	1,2-Diacyl-sn-glycerol (di-isoheptadecanoyl, iso-C17:0)
C00167	UDPglun_c	UDP-glucuronate	C02218	Dha_c	Dehydroalanine	C15650	d3Dik5mylthi1p_c	2,3-Diketo-5-methylthiopentyl-1-phosphate	M00260	tagi170_c	Triacylglycerol (triisoheptadecanoyl, iso-C17:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00169	Carbp_c	Carbamoyl phosphate	C02220	dAmi_c	2-Aminomuconate	C15667	cCar15phoDrbs_c	5-Carboxyamino-1-(5-phospho-D-ribosyl)imidazole	M00261	cdpdiC17g_c	CDP-1,2-diisoheptadecanoylglycerol
C00170	cMylthi_c	5'-Methylthioadenosine	C02226	dMylmal_c	2-Methylmaleate	C15809	Imin_c	Iminoglycine	M00262	u2dgrai150_c	1,2-Diacyl-sn-glycerol (diante-isopentadecanoyl, ante-iso-C15:0)
C00175	Cobaion_c	Cobalt ion	C02232	tOadiCoA_c	3-Oxoadipyl-CoA	C15810	Thib1_c	Thiamine biosynthesis intermediate_1	M00263	tagai150_c	Triacylglycerol (triante-isopentadecanoyl, ante-iso-C15:0)
C00175	Cobalt2_c	Cobalt ion	C02291	LCyst_c	L-Cystathionine	C15811	Enzycys_c	[Enzyme]-cysteine	M00264	cdpdaiC15g_c	CDP-1,2-diante-isopentadecanoylglycerol
C00177	Cyanion_c	Cyanide ion	C02323	Salialco_c	Salicyl alcohol	C15812	EnzySsulylcys_c	[Enzyme]-S-sulfanylcysteine	M00265	u2dgrai170_c	1,2-Diacyl-sn-glycerol (diante-isoheptadecanoyl, ante-iso-C17:0)
C00178	Thymi_c	Thymine	C02336	bDFru_c	beta-D-Fructose	C15813	Thib4_c	Thiamine biosynthesis intermediate_4	M00266	tagai170_c	Triacylglycerol (triante-isoheptadecanoyl, ante-iso-C17:0)
C00179	Agm_c	Agmatine	C02348	RAll_c	(R)(-)-Allantoin	C15814	Thib5_c	Thiamine biosynthesis intermediate_5	M00267	cdpdaiC17g_c	CDP-1,2-diante-isoheptadecanoylglycerol
C00181	DXylo_c	D-Xylose	C02350	SAll_c	(S)-Allantoin	C15972	EnzyN6lip_c	Enzyme N6-(lipoyl)lysine	M00268	u2dgr161_c	1,2-Diacyl-sn-glycerol (dihexadec-9-enoyl, n-C16:1)
C00183	LVal_c	L-Valine	C02412	GlytRNAGly_c	Glycyl-tRNA(Gly)	C15973	EnzyN6dih_c	Enzyme N6-(dihydrolipoyl)lysine	M00269	tag161_c	Triacylglycerol (trihexadec-9-enoyl, n-C16:1)
C00184	Glyce_c	Glycerone	C02463	Prec2_c	Precorrin 2	C15980	S2MylbutoylCoA_c	(S)-2-Methylbutanoyl-CoA	M00270	cdpdhdec9eg_c	CDP-1,2-dihexadec-9-enoylglycerol
C00185	Cell_c	Cellobiose	C02480	cisMuc_c	cis,cis-Muconate	C16219	tOsteoylacp_c	3-Oxostearoyl-[acp]	M00271	u2dgr150_c	1,2-Diacyl-sn-glycerol (dipentadecanoyl, n-C15:0)
C00186	SLac_c	(S)-Lactate	C02501	dHxy_c	2-Hydroxymuconate	C16220	R3Hxyodecaoyl-acp_c	(R)-3-Hydroxyoctadecanoyl-[acp]	M00272	tag150_c	Triacylglycerol (tripentadecanoyl, n-C15:0)
C00188	LThr_c	L-Threonine	C02504	aIsopro_c	alpha-Isopropylmalate	C16221	dEOctoylacp_c	(2E)-Octadecenoyl-[acp]	M00273	cdpdptdecg_c	CDP-1,2-dipentadecanoylglycerol
C00191	DGluc_c	D-Glucuronate	C02505	dPhenylace_c	2-Phenylacetamide	C16242	Cobaprec5A_c	Cobalt-precorrin 5A	M00274	u2dgr170_c	1,2-Diacyl-sn-glycerol (diheptadecanoyl, n-C17:0)
C00194	Cobacoen_c	Cobamide coenzyme	C02593	TdecoylCoA_c	Tetradecanoyl-CoA	C16243	Cobaprec5B_c	Cobalt-precorrin 5B	M00275	tag170_c	Triacylglycerol (triheptadecanoyl, n-C17:0)
C00196	d3Dihbzte_c	2,3-Dihydroxybenzoate	C02612	R2Mylmal_c	(R)-2-Methylmalate	C16244	Cobaprec7_c	Cobalt-precorrin 7	M00276	cdpdhepdecg_c	CDP-1,2-diheptadecanoylglycerol
C00197	tPDgly_c	3-Phospho-D-glycerate	C02631	dIsopro_c	2-Isopropylmaleate	C16254	DihresisuccSsucc_c	[Dihydrolipoyllysine-residue succinyltransferase] S-succinyldihydrolipoyllysine	M00277	pgi140_c	Phosphatidylglycerol (di-isotetradecanoyl, iso-C14:0)
C00198	DGlu15lac_c	D-Glucono-1,5-lactone	C02637	tDhs_c	3-Dehydroshikimate	C16255	DihresiacetSacet_c	[Dihydrolipoyllysine-residue acetyltransferase] S-acetyldihydrolipoyllysine	M00278	clpni140_c	cardiolipin (isotetratetradecanoyl, iso-C14:0)
C00199	DRibu5p_c	D-Ribulose 5-phosphate	C02656	Pim_c	Pimelate	C16269	epiIsoziz_c	(+)-epi-Isozizaene	M00279	pii140_c	phosphatidylinositol (diisotetradecanoyl, iso-C14:0)
C00204	dDh3deoDglu_c	2-Dehydro-3-deoxy-D-gluconate	C02670	DGlulac_c	D-Glucuronolactone	C16286	Geo_c	Geosmin	M00280	psi140_c	phosphatidylserine (diisotetradecanoyl, iso-C14:0)
C00206	dADP_c	dADP	C02679	C120_c	Dodecanoate (n-C12:0)	C16432	cHxy_c	5-Hydroxyectoine	M00281	pei140_c	phosphatidylethanolamine (diisotetradecanoyl, iso-C14:0)
C00207	Aceto_c	Acetone	C02700	LFry_c	L-Formylkynurenine	C16466	steMyl3o6octoyl-CoA_c	7-Methyl-3-oxo-6-octenoyl-CoA	M00282	pgi160_c	Phosphatidylglycerol (diisohexadecanoyl, iso-C16:0)
C00208	Mal_c	Maltose	C02714	NAceput_c	N-Acetylputrescine	C16468	dE5Mylhex24dieoylCoA_c	(2E)-5-Methylhexa-2,4-dienoyl-CoA	M00283	clpni160_c	cardiolipin (isotetrahexadecanoyl, iso-C16:0)
C00209	Oxala_c	Oxalate	C02730	Succylbenzoate_c	2-Succinylbenzoate	C16469	tHxy5mylhex4enoylCoA_c	3-Hydroxy-5-methylhex-4-enoyl-CoA	M00284	pii160_c	phosphatidylinositol (diisohexadecanoyl, iso-C16:0)
C00212	Aden_c	Adenosine	C02737	Pti_c	Phosphatidylserine	C16470	cMylhex4enooyl-CoA_c	5-Methylhex-4-enoyl-CoA	M00285	psi160_c	phosphatidylserine (diisohexadecanoyl, iso-C16:0)
C00213	Sar_c	Sarcosine	C02739	u5PDrbsATP_c	1-(5-Phospho-D-ribosyl)-ATP	C16471	cMyl3o4hexoyl-CoA_c	5-Methyl-3-oxo-4-hexenoyl-CoA	M00286	pgi180_c	Phosphatidylglycerol (di-isooctadecanoyl, iso-C18:0)
C00214	Thym_c	Thymidine	C02741	PrbAMP_c	Phosphoribosyl-AMP	C16519	Succyl5enolpyrylhxy3cyne1carlate_c	2-Succinyl-5-enolpyruvyl-6-hydroxy-3-cyclohexene-1-carboxylate	M00287	clpni180_c	cardiolipin (isotetraoctadecanoyl, iso-C18:0)
C00217	DGluta_c	D-Glutamate	C02876	Propp_c	Propanoyl phosphate	C16520	Hexaoylacp_c	Hexadecenoyl-[acyl-carrier protein]	M00288	pii180_c	phosphatidylinositol (di-isooctadecanoyl, iso-C18:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00221	bDGlu_c	beta-D-Glucose	C02939	tMylbutoylCoA_c	3-Methylbutanoyl-CoA	C16537	ptdca_c	Pentadecanoate (n-C15:0)	M00289	psi180_c	phosphatidylserine (diisooctadecanoyl, iso-C18:0)
C00222	tOprote_c	3-Oxopropanoate	C02946	qActrte_c	4-Acetamidobutanoate	C16698	NAcetmuraad6p_c	N-Acetylmuramic acid 6-phosphate	M00290	pei160_c	phosphatidylethanolamine (diisohexadecanoyl, iso-C16:0)
C00224	Adensulf_c	Adenylyl sulfate	C02946	qAcete_c	4-Acetamidobutanoate	C16737	cDxDglu_c	5-Deoxy-D-glucuronate	M00291	pei180_c	phosphatidylethanolamine (diisooctadecanoyl, iso-C18:0)
C00227	Acetp_c	Acetyl phosphate	C02967	DNA5mylcyto_c	DNA 5-methylcytosine	C17023	Sulfdono_c	Sulfur donor	M00292	pgi150_c	Phosphatidylglycerol (diisopentadecanoyl, iso-C15:0)
C00229	ACP_c	Acyl-carrier protein	C02972	Dihoyl_c	Dihydrolipoylprotein	C17234	dAmi2enote_c	2-Aminobut-2-enoate	M00293	clpni150_c	cardiolipin (isotetrapentadecanoyl, iso-C15:0)
C00230	t4Dihbzte_c	3,4-Dihydroxybenzoate	C02987	LGlutRNAGlu_c	L-Glutamyl-tRNA(Glu)	C17401	CobafactIII_c	Cobalt-factor III	M00294	pii150_c	phosphatidylinositol (diisopentadecanoyl, iso-C15:0)
C00231	DXylu5p_c	D-Xylulose 5-phosphate	C02995	Mal6p_c	Maltose 6'-phosphate	C17541	UndnyldipNactoyl-NaceLalanylDiso-glunylLlysgl-DalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-D-isoglutaminyl-L-lysyl-(glycyl)-D-alanyl-D-alanine	M00295	psi150_c	phosphatidylserine (diisopentadecanoyl, iso-C15:0)
C00232	Succsald_c	Succinate semialdehyde	C03069	tMylcronylCoA_c	3-Methylcrotonyl-CoA	C17542	UndnyldipNactoyl-NaceLalanylDiso-glunylLlysglycyl-3DalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-D-isoglutaminyl-L-lysyl-(glycyl)3-D-alanyl-D-alanine	M00296	pei150_c	phosphatidylethanolamine (diisopentadecanoyl, iso-C15:0)
C00233	qMyl2opete_c	4-Methyl-2-oxopentanoate	C03082	qPLasp_c	4-Phospho-L-aspartate	C17549	UndnyldipNactoyl-NaceLalanylgam-DgluLlysLalanyl-LalanylDalanyl-Dala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-gamma-D-glutamyl-L-lysyl-(L-alanyl-L-alanyl)-D-alanyl-D-alanine	M00297	pgi170_c	Phosphatidylglycerol (diisoheptadecanoyl, iso-C17:0)
C00234	dFryhfol_c	10-Formyltetrahydrofolate	C03089	cMylthiDrib_c	5-Methylthio-D-ribose	C17550	UndnyldipNactoyl-NaceLalanylgam-DgluLlysLalanyl-DalanylDala_c	Undecaprenyl-diphospho-N-acetylmuramoyl-(N-acetylglucosamine)-L-alanyl-gamma-D-glutamyl-L-lysyl-(L-alanyl)-D-alanyl-D-alanine	M00298	clpni170_c	cardiolipin (isotetraheptadecanoyl, iso-C17:0)
C00235	dmylallydp_c	Dimethylallyl diphosphate	C03090	cPrbs_c	5-Phosphoribosylamine	C17556	ditrancisUndep_c	di-trans,poly-cis-Undecaprenyl phosphate	M00299	pii170_c	phosphatidylinositol (diisoheptadecanoyl, iso-C17:0)
C00236	tPDglycp_c	3-Phospho-D-glyceroyl phosphate	C03114	dmyben_c	Dimethylbenzimidazole	C17954	Alb_c	Albaflavenone	M00300	psi170_c	phosphatidylserine (diisoheptadecanoyl, iso-C17:0)
C00237	Cmonoxide_c	Carbon_monoxide	C03150	Nicbrib_c	Nicotinamide-beta-riboside	C18026	dSEthnylCoA_c	(2S)-Ethylmalonyl-CoA	M00301	pei170_c	phosphatidylethanolamine (diisoheptadecanoyl, iso-C17:0)
C00238	K_c	Potassium cation	C03160	SuccylbenylCoA_c	2-Succinylbenzoyl-CoA	C18028	LFuc15lac_c	L-Fucono-1,5-lactone	M00302	pgai150_c	Phosphatidylglycerol (diante-isopentadecanoyl, ante-iso-C15:0)
C00239	dCMP_c	dCMP	C03175	Shi3p_c	Shikimate 3-phosphate	C18324	dSMylsucnylCoA_c	(2S)-Methylsuccinyl-CoA	M00303	clpnai150_c	cardiolipin (ante-isotetrapentadecanoyl, ante-iso-C15:0)
C00242	Gua_c	Guanine	C03194	R1Ami2ol_c	(R)-1-Aminopropan-2-ol	C19152	CoenF4201_c	Coenzyme F420-1	M00304	piai150_c	phosphatidylinositol (diante-isopentadecanoyl, ante-iso-C15:0)
C00243	Lac_c	Lactose	C03221	dtraDodoylCoA_c	2-trans-Dodecenoyl-CoA	C19153	CoenF4200_c	Coenzyme F420-0	M00305	psai150_c	phosphatidylserine (diante-isopentadecanoyl, ante-iso-C15:0)
C00244	Nitr_c	Nitrate	C03227	tHxyLkyn_c	3-Hydroxy-L-kynurenine	C19154	ste8Did8hyd5dea_c	7,8-Didemethyl-8-hydroxy-5-deazariboflavin	M00306	peai150_c	phosphatidylethanolamine (diante-isopentadecanoyl, ante-iso-C15:0)
C00249	Hexaad_c	Hexadecanoic acid	C03231	tMylglunylCoA_c	3-Methylglutaconyl-CoA	C19155	dSLac2dip5gua_c	(2S)-Lactyl-2-diphospho-5'-guanosine	M00307	pgai170_c	Phosphatidylglycerol (diante-isoheptadecanoyl, ante-iso-C17:0)
C00251	Cho_c	Chorismate	C03232	tPno_c	3-Phosphonooxypyruvate	C19156	dS2Pla_c	(2S)-2-Phospholactate	M00308	clpnai170_c	cardiolipin (ante-isotetraheptadecanoyl, ante-iso-C17:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00253	Nico_c	Nicotinate	C03263	CoprIII_c	Coproporphyrinogen III	C19673	Malnmyleste_c	Malonyl-[acp] methyl ester	M00309	piai170_c	phosphatidylinositol (diante-isoheptadecanoyl, ante-iso-C17:0)
C00254	Pre_c	Prephenate	C03283	L24Diate_c	L-2,4-Diaminobutanoate	C19845	Pimoylacp_c	Pimeloyl-[acyl-carrier protein]	M00310	psai170_c	phosphatidylserine (diante-isoheptadecanoyl, ante-iso-C17:0)
C00255	Rib_c	Riboflavin	C03287	LGlut5p_c	L-Glutamyl 5-phosphate	C19846	Pimeacpmyleste_c	Pimeloyl-[acyl-carrier protein] methyl ester	M00311	peai170_c	phosphatidylethanolamine (diante-isoheptadecanoyl, ante-iso-C17:0)
C00256	RLac_c	(R)-Lactate	C03289	LxylHexlac_c	L-xylo-Hexulonolactone	C19945	tO56dehCoA_c	3-Oxo-5,6-dehydrosuberyl-CoA	M00312	pg161_c	Phosphatidylglycerol (dihexadec-9-enoyl, n-C16:1)
C00257	DGlucad_c	D-Gluconic acid	C03298	NaMylhis_c	Nalpha-Methylhistidine	C19946	tO56dehyCoAsald_c	3-Oxo-5,6-dehydrosuberyl-CoA semialdehyde	M00313	clpn161_c	cardiolipin (tetrahexadec-9-enoyl, n-C16:1)
C00258	DGly_c	D-Glycerate	C03344	dMylactCoA_c	2-Methylacetoacetyl-CoA	C19975	dOxe23HyliCoA_c	2-Oxepin-2(3H)-ylideneacetyl-CoA	M00314	pi161_c	phosphatidylinositol (dihexadec-9enoyl, n-C16:1)
C00259	LAra_c	L-Arabinose	C03345	dMylbut2enooyl-CoA_c	2-Methylbut-2-enoyl-CoA	C20062	d12Epo12dihnyl-CoA_c	2-(1,2-Epoxy-1,2-dihydrophenyl)acetyl-CoA	M00315	ps161_c	phosphatidylserine (dihexadec-9-enoyl, n-C16:1)
C00262	Hyp_c	Hypoxanthine	C03373	Amiiribo_c	Aminoimidazole ribotide	C20237	aMalt1p_c	alpha-Maltose 1-phosphate	M00316	pe161_c	phosphatidylethanolamine (dihexadec-9enoyl, n-C16:1)
C00263	LHomo_c	L-Homoserine	C03406	NLArg_c	N-(L-Arginino)succinate	C20238	dREthnylCoA_c	(2R)-Ethylmalonyl-CoA	M00317	pg150_c	Phosphatidylglycerol (dipentadecanoyl, n-C15:0)
C00266	Glylde_c	Glycolaldehyde	C03453	gamOxa_c	gamma-Oxalocrotonate	C20246	d2R5Z2Carb4mylthia52Hylidp_c	2-[(2R,5Z)-2-Carboxy-4-methylthiazol-5(2H)-ylidene]ethyl phosphate	M00318	clpn150_c	cardiolipin (tetrapentadecanoyl, n-C15:0)
C00267	aDGlu_c	alpha-D-Glucose	C03460	dMylpro2enooyl-CoA_c	2-Methylprop-2-enoyl-CoA	C20247	d2Carb4mylthia5-ylep_c	2-(2-Carboxy-4-methylthiazol-5-yl)ethyl phosphate	M00319	pi150_c	phosphatidylinositol (dipentadecanoyl, n-C15:0)
C00269	CDPdia_c	CDP-diacylglycerol	C03479	Foliad_c	Folinic acid	C20258	dS4S4Hxy2345-thd_c	(2S,4S)-4-Hydroxy-2,3,4,5-tetrahydrodipicolinate	M00320	ps150_c	phosphatidylserine (dipentadecanoyl, n-C15:0)
C00270	NAceneu_c	N-Acetylneuraminate	C03492	D4Ppa_c	D-4'-Phosphopantothenate	C20372	tKetoacpmyleste_c	3-Ketoglutaryl-[acp] methyl ester	M00321	pe150_c	phosphatidylethanolamine (dipentadecanoyl, n-C15:0)
C00275	DMann6p_c	D-Mannose 6-phosphate	C03506	Indop_c	Indoleglycerol phosphate	C20373	tHxygacpmyleste_c	3-Hydroxyglutaryl-[acp] methyl ester	M00322	pg170_c	Phosphatidylglycerol (diheptadecanoyl, n-C17:0)
C00279	DEryt4p_c	D-Erythrose 4-phosphate	C03508	L2Ami3obutad_c	L-2-Amino-3-oxobutanoic acid	C20374	Enoyacpmyleste_c	Enoylglutaryl-[acp] methyl ester	M00323	clpn170_c	cardiolipin (tetraheptadecanoyl, n-C17:0)
C00283	Hgensulf_c	Hydrogen sulfide	C03518	NAcebDglu_c	N-Acetyl-beta-D-glucosaminide	C20375	Glutacpmyleste_c	Glutaryl-[acp] methyl ester	M00324	pi170_c	phosphatidylinositol (diheptadecanoyl, n-C17:0)
C00286	dGTP_c	dGTP	C03561	R3HxyoylCoA_c	(R)-3-Hydroxybutanoyl-CoA	C20376	tKetopacpmylester_c	3-Ketopimeloyl-[acp] methyl ester	M00325	ps170_c	phosphatidylserine (diheptadecanoyl, n-C17:0)
C00288	HCO3_c	HCO3-	C03586	dO23dih5ace_c	2-Oxo-2,3-dihydrofuran-5-acetate	C20377	tHxypacpmyleste_c	3-Hydroxypimeloyl-[acp] methyl ester	M00326	pe170_c	phosphatidylethanolamine (diheptadecanoyl, n-C17:0)
C00291	Ni_c	Nickel	C03589	qHxy2opete_c	4-Hydroxy-2-oxopentanoate	C20378	Enoylacpmylester_c	Enoylpimeloyl-[acp] methyl ester	M00327	pgpi140_c	Phosphatidylglycerophosphate (diisotetradecanoyl, iso-C14:0)
C00294	Ino_c	Inosine	C03656	S5Ami3ohexad_c	(S)-5-Amino-3-oxohexanoic acid	C20589	DGluco6p_c	D-Glucosaminate-6-phosphate	M00328	pgpi160_c	Phosphatidylglycerophosphate (diisohexadecanoyl, iso-C16:0)
C00295	Oro_c	Orotate	C03657	Dhxy2naphthoate_c	1,4-Dihydroxy-2-naphthoate	C20781	d4Dik3deoLfuc_c	2,4-Diketo-3-deoxy-L-fuconate	M00329	pgpi180_c	Phosphatidylglycerophosphate (diisooctadecanoyl, iso-C18:0)
C00296	Quin_c	Quinate	C03680	qImi5prote_c	4-Imidazolone-5-propanoate	C20889	DGal15lac_c	D-Galactaro-1,5-lactone	M00330	pgpi150_c	Phosphatidylglycerophosphate (diisopentadecanoyl, iso-C15:0)
C00299	Uri_c	Uridine	C03722	Quinol_c	Quinolinate	C20896	DGal14lac_c	D-Galactaro-1,4-lactone	M00331	pgpi170_c	Phosphatidylglycerophosphate (diisoheptadecanoyl, iso-C17:0)
C00302	Gluta_c	Glutamate	C03741	S4Ami5opete_c	(S)-4-Amino-5-oxopentanoate	C20994	SHer2ylLcystSoxid_c	S-(Hercyn-2-yl)-L-cysteine S-oxide	M00332	pgpai150_c	Phosphatidylglycerophosphate (diante-isopentadecanoyl, ante-iso-C15:0)
C00305	Mg2_c	Magnesium cation	C03752	dAmi2deoDglu_c	2-Amino-2-deoxy-D-gluconate	C20995	gammLGlutSher2yl-LcystSoxid_c	gamma-L-Glutamyl-S-(hercyn-2-yl)-L-cysteine S-oxide	M00333	pgpai170_c	Phosphatidylglycerophosphate (diante-isoheptadecanoyl, ante-iso-C17:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00310	DXyl_c	D-Xylulose	C03785	DTaga16bp_c	D-Tagatose 1,6-bisphosphate	M00001	propnylACP_c	PropionylACP	M00334	pgp161_c	Phosphatidylglycerophosphate (dihexadec-9-enoyl, n-C16:1)
C00311	Isocit_c	Isocitrate	C03794	N612DicAMP_c	N6-(1,2-Dicarboxyethyl)-AMP	M00002	pentadecanoyl-ACP_c	PentadecanoylACP	M00335	pgp150_c	Phosphatidylglycerophosphate (dipentadecanoyl, n-C15:0)
C00320	Thio_c	Thiosulfate	C03824	dAmimsald_c	2-Aminomuconate semialdehyde	M00003	heptadecanoyl-ACP_c	HeptadecanoylACP	M00336	pgp170_c	Phosphatidylglycerophosphate (diheptadecanoyl, n-C17:0)
C00322	dOadi_c	2-Oxoadipate	C03827	dDh3deoL_fuc_c	2-Dehydro-3-deoxy-L-fuconate	M00004	uAcylsngly3p_m-_m_c	uAcylsngly3p_m	M00337	iC13p_c	Isotridecanoyl-phosphate (iso-C13:0)
C00327	LCit_c	L-Citrulline	C03838	cPrb_c	5'-Phosphoribosylglycinamide	M00005	Phos_m_c	Phos_m	M00338	aiC13p_c	Anteisotridecanoyl-phosphate (ante-iso-C13:0)
C00328	LKyn_c	L-Kynurenine	C03892	Ptidylglycerop_c	Phosphatidylglycerophosphate	M00006	Orth_m_c	Orthophosphate	M00339	aiC171p_c	Anteisoheptadecenoyl-phosphate (ante-iso-C17:1)
C00330	Dxg_c	Deoxyguanosine	C03912	S1Pyr5car_c	(S)-1-Pyrroline-5-carboxylate	M00007	CDPdiac_m_c	CDPdiac_m	M00340	iC171p_c	Isoheptadecenoyl-phosphate (iso-C17:1)
C00332	ActCoA_c	Acetoacetyl-CoA	C03921	dDh3deoDglu_c	2-Dehydro-3-deoxy-D-glucarate	M00008	Ptidylseri_m_c	Ptidylseri_m	M00341	C171p_c	heptadecenoyl-phosphate (C17:1)
C00333	DGala_c	D-Galacturonate	C03939	Aceacp_c	Acetyl-[acyl-carrier protein]	M00009	Biomass_c	Biomass_biosynthesis	M00342	C12p_c	Dodecanoly-phosphate (n-C12:0)
C00334	qAmite_c	4-Aminobutanoate	C03972	d345Thy_c	2,3,4,5-Tetrahydrodipicolinate	M00010	Thead_c	Thead	M00343	C14p_c	Tetradecanoyl-phosphate (n-C14:0)
C00337	SDih_c	(S)-Dihydroorotate	C04006	uDmyoInos3p_c	1D-myo-Inositol 3-phosphate	M00011	DNA_bio_c	DNA_biosynthesis	M00344	C16p_c	Hexadecanoyl-phosphate (n-C16:0)
C00341	Geradp_c	Geranyl diphosphate	C04039	d3Dih3mylbutte_c	2,3-Dihydroxy-3-methylbutanoate	M00012	RNA_bio_c	RNA_biosynthesis	M00345	C18p_c	Octadecanoyl-phosphate (n-C18:0)
C00342	Thi_c	Reduced thioredoxin	C04079	RPttenoylLcys_c	N-((R)-Pantothenoyl)-L-cysteine	M00013	Protein_c	Protein_biosynthesis	M00346	pai130_c	1,2-diisotridecanoyl-sn-glycerol 3-phosphate
C00343	Thiodisu_c	Oxidized thioredoxin	C04088	Octdecoylacp_c	Octadecanoyl-[acyl-carrier protein]	M00014	Plipid_c	Phospholipid_biosynthesis	M00347	paai130_c	1,2-dianteisotridecanoyl-sn-glycerol 3-phosphate
C00344	Ptidy_c	Phosphatidylglycerol	C04122	D1Amip2olOp_c	D-1-Aminopropan-2-ol O-phosphate	M00015	TAG_c	TAG_biosynthesis	M00348	paai171_c	1,2-dianteisoheptadecenoyl-sn-glycerol 3-phosphate
C00345	sPDglu_c	6-Phospho-D-gluconate	C04133	NAcetLglut5p_c	N-Acetyl-L-glutamate 5-phosphate	M00016	Smallmol_c	Smallmol_pool_biosynthesis	M00349	pai171_c	1,2-diisoheptadecenoyl-sn-glycerol 3-phosphate
C00346	Ethap_c	Ethanolamine phosphate	C04146	alltOctnyldp_c	all-trans-Octaprenyl diphosphate	M00017	Peptidoglycan_c	Peptidoglycan_biosynthesis	M00350	pa171_c	1,2-diheptadecenoyl-sn-glycerol 3-phosphate
C00350	Ptidyletholamne_c	Phosphatidylethanolamine	C04180	cdec3eACP_c	cis-dec-3-enoyl-[acyl-carrier protein] (n-C10:1)	M00018	Carbohyd_c	Carbohydrates_biosynthesis	M00351	pa120_c	1,2-didodecanoyl-sn-glycerol 3-phosphate
C00352	DGluc6p_c	D-Glucosamine 6-phosphate	C04181	tHxy3myl2obutad_c	3-Hydroxy-3-methyl-2-oxobutanoic acid	M00019	Nhxycad_c	N-hydroxycadaverine	M00352	pa140_c	1,2-ditetradecanoyl-sn-glycerol 3-phosphate
C00353	Gerandp_c	Geranylgeranyl diphosphate	C04188	SMyl5thioDribo-1p_c	S-Methyl-5-thio-D-ribose 1-phosphate	M00020	NhxyNsuccylcad_c	N-hydroxy-N-succinylcadaverine	M00353	pa160_c	1,2-dihexadecanoyl-sn-glycerol 3-phosphate
C00354	DFruc16bp_c	D-Fructose 1,6-bisphosphate	C04216	alltHeptyldp_c	all-trans-Heptaprenyl diphosphate	M00021	DesfE_c	Desferrioxamine-E	M00354	pa180_c	1,2-dioctadecanoyl-sn-glycerol 3-phosphate
C00355	t4DihLphenyl_c	3,4-Dihydroxy-L-phenylalanine	C04236	dS2Isopro3osu_c	(2S)-2-Isopropyl-3-oxosuccinate	M00022	DesfG_c	Desferrioxamine-G	M00355	uiC13g3p_c	1-isotridecanoyl-sn-glycerol 3-phosphate
C00356	S3Hxy3mylglr-CoA_c	(S)-3-Hydroxy-3-methylglutaryl-CoA	C04246	But2enooylacp_c	But-2-enoyl-[acyl-carrier protein]	M00023	NactylNhxycad_c	N-acetyl-N-hydroxycadaverine	M00356	uaiC13g3p_c	1-ante-isotridecanoyl-sn-glycerol 3-phosphate
C00357	NAce6p_c	N-Acetyl-D-glucosamine 6-phosphate	C04253	Electranflav_c	Electron-transferring flavoprotein	M00024	DesfB_c	Desferrioxamine-B	M00357	uaiC171g3p_c	1-ante-isoheptadecenoyl-sn-glycerol 3-phosphate
C00360	dAMP_c	dAMP	C04259	NNdmyhis_c	N,N-Dimethylhistidine	M00025	Prochax_c	Prochaxamycin	M00358	uiC171g3p_c	1-isoheptadecenoyl-sn-glycerol 3-phosphate
C00361	dGDP_c	dGDP	C04261	ProtNpiphosLhist_c	Protein N(pi)-phospho-L-histidine	M00026	MethylDonor_c	Methyl_Donor	M00359	uC171g3p_c	1-heptadecenoyl-sn-glycerol 3-phosphate
C00362	dGMP_c	dGMP	C04272	R23Dih3mylbutte_c	(R)-2,3-Dihydroxy-3-methylbutanoate	M00027	ChaxA_c	Chaxamycin-A	M00360	uC12g3p_c	1-dodecanoyl-sn-glycerol 3-phosphate
C00363	dTDP_c	dTDP	C04281	L1Pyr3hyd5car_c	L-1-Pyrroline-3-hydroxy-5-carboxylate	M00028	ChaxB_c	Chaxamycin-B	M00361	uC14g3p_c	1-tetradecanoyl-sn-glycerol 3-phosphate
C00364	dTMP_c	dTMP	C04287	tD354Tri12dio_c	3D-3,5/4-Trihydroxycyclohexane-1,2-dione	M00029	tetraketideCxm_c	tetraketide_intermediate_Cxm	M00362	uC16g3p_c	1-hexadecanoyl-sn-glycerol 3-phosphate
C00365	dUMP_c	dUMP	C04302	N5PDrbs_c	N-(5-Phospho-D-ribosyl)anthranilate	M00030	m_tetraketideCxm_c	mod tetraketide_intermediate_Cxm	M00363	uC18g3p_c	1-octadecanoyl-sn-glycerol 3-phosphate
C00366	Urate_c	Urate	C04332	s7dmy8Drib_c	6,7-Dimethyl-8-(D-ribityl)lumazine	M00031	heptaketideCxm_c	heptaketide_intermediate_Cxm	M00364	u2dgri130_c	1,2-Diacyl-sn-glycerol (diisotridecanoyl, iso-C13:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00369	Star_c	Starch	C04352	R4PpaoylLcys_c	(R)-4'-Phosphopantothenoyl-L-cysteine	M00032	octaketideCxm_c	octaketide_intermediate_Cxm	M00365	u2dgrai130_c	1,2-Diacyl-sn-glycerol (dianteisotridecanoyl, ante-iso-C13:0)
C00369_	Star_c	Starch	C04376	cPrbNfor_c	5'-Phosphoribosyl-N-formylglycinamide	M00033	nonaketideCxm_c	nonaketide_intermediate_Cxm	M00366	u2dgrai171_c	1,2-Diacyl-sn-glycerol (dianteisoheptadecenoyl, ante-iso-C17:1)
C00380	Cyto_c	Cytosine	C04377	c10Mylene_c	5,10-Methylenetetrahydromethanopterin	M00034	undecaketideCxm_c	undecaketide_intermediate_Cxm	M00367	u2dgri171_c	1,2-Diacyl-sn-glycerol (diisoheptadecenoyl, iso-C17:1)
C00385	Xan_c	Xanthine	C04405	dS3S3Hxy2mylbutoylCoA_c	(2S,3S)-3-Hydroxy-2-methylbutanoyl-CoA	M00035	t4ketideCxl_c	tetraketide_intermediate_Cxl	M00368	u2dgr171_c	1,2-Diacyl-sn-glycerol (diheptadecenoyl, C17:1)
C00387	Guan_c	Guanosine	C04409	dAmi3carbsald_c	2-Amino-3-carboxymuconate semialdehyde	M00036	h6ketideCxl_c	hexaketide_intermediate_Cxl	M00369	u2dgr120_c	1,2-Diacyl-sn-glycerol (didodecanoyl, n-C12:0)
C00390	Ubinol_c	Ubiquinol	C04411	dR3S3Isopro_c	(2R,3S)-3-Isopropylmalate	M00037	o8ketideCxl_c	octaketide_intermediate_Cxl	M00370	u2dgr140_c	1,2-Diacyl-sn-glycerol (ditetradecanoyl, n-C14:0)
C00399	Ubi_c	Ubiquinone	C04419	Carbcarbcarrprot_c	Carboxybiotin-carboxyl-carrier protein	M00038	d10ketideCxl_c	decaketide_intermediate_Cxl	M00371	u2dgr160_c	1,2-Diacyl-sn-glycerol (dihexadecanoyl, n-C16:0)
C00402	DAsp_c	D-Aspartate	C04421	NSucnylLL26diate_c	N-Succinyl-LL-2,6-diaminoheptanedioate	M00039	CxlA_c	Chaxalactin_A	M00372	u2dgr180_c	1,2-Diacyl-sn-glycerol (dioctadecanoyl, n-C18:0)
C00404	NPolyp_c	Polyphosphate	C04425	SAde4mylthi2obute_c	S-Adenosyl-4-methylthio-2-oxobutanoate	M00040	peptideHal_int_c	peptideHal_intermediate	M00373	tagi130_c	Triacylglycerol (triisotridecanoyl, iso-C13:0)
C00404	N_1Polyp_c	Polyphosphate-1	C04442	dDh3deo6phoDglu_c	2-Dehydro-3-deoxy-6-phospho-D-gluconate	M00041	dipephexketide_c	dipeptide_hexaketide	M00374	tagai130_c	Triacylglycerol (trianteisotridecanoyl, ante-iso-C13:0)
C00407	LIsoleu_c	L-Isoleucine	C04454	cAmi65phoDrib_c	5-Amino-6-(5'-phospho-D-ribitylamino)uracil	M00042	proPKNP_c	pro hybrid PKNP	M00375	tagai171_c	Triacylglycerol (trianteisoheptadecenoyl, ante-iso-C17:1)
C00409	Met_c	Methanethiol	C04462	NSucnyl2Lami6ohete_c	N-Succinyl-2-L-amino-6-oxoheptanedioate	M00043	isoC141ACP_c	Isobutyryl-ACP (iso-C4:0ACP)	M00376	tagi171_c	Triacylglycerol (triisoheptadecenoyl, iso-C17:1)
C00415	Dih_c	Dihydrofolate	C04494	Guan3dp5trip_c	Guanosine 3'-diphosphate 5'-triphosphate	M00044	t3oiC6ACP_c	3-Oxoisohexanoyl-[acyl-carrier protein]	M00377	tag171_c	Triacylglycerol (triheptadecenoyl, n-C17:1)
C00416	Ptida_c	Phosphatidate	C04501	NAcetaDgluc1p_c	N-Acetyl-alpha-D-glucosamine 1-phosphate	M00045	t3hiC6ACP_c	(R)-3-Hydroxyisohexanoyl-[acyl-carrier protein]	M00378	tag120_c	Triacylglycerol (tridodecanoyl, n-C12:0)
C00417	cisAco_c	cis-Aconitate	C04556	qAmi2myl5phopyr_c	4-Amino-2-methyl-5-phosphomethylpyrimidine	M00046	tiC62eACP_c	trans-isohex-2-enoyl-[acyl-carrier protein]	M00379	tag140_c	Triacylglycerol (tritetradecanoyl, n-C14:0)
C00418	RMev_c	(R)-Mevalonate	C04570	Reduelectranflav_c	Reduced electron-transferring flavoprotein	M00047	iC6ACP_c	Isohexanoyl-ACP (iso-C6:0ACP)	M00380	tag160_c	Triacylglycerol (trihexadecanoyl, n-C16:0)
C00424	SLaclde_c	(S)-Lactaldehyde	C04574	ditrancisUndedp_c	di-trans,poly-cis-Undecaprenyl diphosphate	M00048	t3oiC8ACP_c	3-Oxoisooctanoyl-[acyl-carrier protein]	M00381	tag180_c	Triacylglycerol (trioctadecanoyl, n-C18:0)
C00430	cAmi_c	5-Aminolevulinate	C04582	SMyl5thioDribu1p_c	S-Methyl-5-thio-D-ribulose 1-phosphate	M00049	t3hiC8ACP_c	(R)-3-Hydroxyisooctanoyl-[acyl-carrier protein]	M00382	cdpdiC13g_c	CDP-1,2-diisotridecanoylglycerol
C00433	d5Diote_c	2,5-Dioxopentanoate	C04618	tR3Hxyoylacp_c	(3R)-3-Hydroxybutanoyl-[acyl-carrier protein]	M00050	tiC82eACP_c	trans-isooct-2-enoyl-[acyl-carrier protein]	M00383	cdpdaiC13g_c	CDP-1,2-dianteisotridecanoylglycerol
C00436	NCaroyl_c	N-Carbamoylputrescine	C04619	tR3Hxydecoylacp_c	(3R)-3-Hydroxydecanoyl-[acyl-carrier protein]	M00051	iC8ACP_c	Isooctanoyl-ACP (iso-C8:0ACP)	M00384	cdpdaiC171g_c	CDP-1,2-dianteisoheptadecenoylglycerol
C00437	NAceorn_c	N-Acetylornithine	C04620	tR3Hxyooylacp_c	(3R)-3-Hydroxyoctanoyl-[acyl-carrier protein]	M00052	t3oiC10ACP_c	3-Oxoisodecanoyl-[acyl-carrier protein]	M00385	cdpdiC171g_c	CDP-1,2-diisoheptadecenoylglycerol
C00438	NCaroylLasp_c	N-Carbamoyl-L-aspartate	C04631	UDPNace31carnylDglu_c	UDP-N-acetyl-3-(1-carboxyvinyl)-D-glucosamine	M00053	t3hiC10ACP_c	(R)-3-Hydroxyisodecanoyl-[acyl-carrier protein]	M00386	cdpdC17g_c	CDP-1,2-diheptadecenoylglycerol
C00439	NForLglu_c	N-Formimino-L-glutamate	C04633	tR3Hxypoylacp_c	(3R)-3-Hydroxypalmitoyl-[acyl-carrier protein]	M00054	tiC102eACP_c	trans-isodec-2-enoyl-[acyl-carrier protein]	M00387	cdpdC12g_c	CDP-1,2-didodecanoylglycerol
C00440	cMylthfol_c	5-Methyltetrahydrofolate	C04640	dFoN15pho_c	2-(Formamido)-N1-(5'-phosphoribosyl)acetamidine	M00055	iC10ACP_c	Isodecanoyl-ACP (iso-C10:0ACP)	M00388	cdpdC14g_c	CDP-1,2-ditetradecanoylglycerol
C00441	LAsp4sald_c	L-Aspartate 4-semialdehyde	C04666	Deryt1Imi4ylg3p_c	D-erythro-1-(Imidazol-4-yl)glycerol 3-phosphate	M00056	t3oiC12ACP_c	3-Oxoisododecanoyl-[acyl-carrier protein]	M00389	cdpdC16g_c	CDP-1,2-dihexadecanoylglycerol
C00445	c10Metnylhfol_c	5,10-Methenyltetrahydrofolate	C04675	tHxy34mylpen3en1ylCoA_c	3-Hydroxy-3-(4-methylpent-3-en-1-yl)glutaryl-CoA	M00057	t3hiC12ACP_c	(R)-3-Hydroxyisododecanoyl-[acyl-carrier protein]	M00390	cdpdC18g_c	CDP-1,2-dioctadecanoylglycerol
C00446	aDGa1p_c	alpha-D-Galactose 1-phosphate	C04677	u5Prb5ami4imi_c	1-(5'-Phosphoribosyl)-5-amino-4-imidazolecarboxamide	M00058	tiC122eACP_c	trans-isododec-2-enoyl-[acyl-carrier protein]	M00391	clpniC130_c	cardiolipin (tetraisotridecanoyl, iso-C13:0)
C00448	tranFarndp_c	trans,trans-Farnesyl diphosphate	C04688	tR3Hxytdecaoylacp_c	(3R)-3-Hydroxytetradecanoyl-[acyl-carrier protein]	M00059	iC12ACP_c	Isododecanoyl-ACP (iso-C12:0ACP)	M00392	clpnaiC130_c	cardiolipin (tetraanteisotridecanoyl, ante-iso-C13:0)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00455	NicoDribo_c	Nicotinamide D-ribonucleotide	C04691	dDh3deoxDarabhept7P_c	7-Dehydro-3-deoxy-D-arabino-heptonate 7-phosphate	M00060	t3oiC14ACP_c	3-Oxoisotetradecanoyl-[acyl-carrier protein]	M00393	clpnaiC171_c	cardiolipin (tetraanteisoheptadecenoyl, ante-iso-C17:1)
C00458	dCTP_c	dCTP	C04702	UDPoylLAlaDgamGluLLysDAlaDAla_c	UDPMurNAc(oyl-L-Ala-D-gamma-Glu-L-Lys-D-Ala-D-Ala)	M00061	t3hiC14ACP_c	(R)-3-Hydroxyisotetradecanoyl-[acyl-carrier protein]	M00394	clpniC171_c	cardiolipin (tetraisoheptadecenoyl, iso-C17:1)
C00459	dTTP_c	dTTP	C04732	cAmi61Drib_c	5-Amino-6-(1-D-ribitylamino)uracil	M00062	tiC142eACP_c	trans-isoetradec-2-enoyl-[acyl-carrier protein]	M00395	clpnC171_c	cardiolipin (tetraheptadecenoyl, n-C17:1)
C00460	dUTP_c	dUTP	C04734	u5Prb5for4imi_c	1-(5'-Phosphoribosyl)-5-formamido-4-imidazolecarboxamide	M00063	iC14ACP_c	Isomyristoyl-ACP (iso-C14:0ACP)	M00396	clpnC120_c	cardiolipin (tetradodecanoyl, n-C12:0)
C00461	Chit_c	Chitin	C04751	u5PDrbs5ami4imi_c	1-(5-Phospho-D-ribosyl)-5-amino-4-imidazolecarboxylate	M00064	t3oiC16ACP_c	3-Oxoisohexadecanoyl-[acyl-carrier protein]	M00397	clpnC140_c	cardiolipin (tetratetradecanoyl, n-C14:0)
C00461	Chit__c	Chitin	C04752	dMyl4amin5hydrpyridp_c	2-Methyl-4-amino-5-hydroxymethylpyrimidine diphosphate	M00065	t3hiC16ACP_c	R-3-hydroxyisopalmitoyl-[acyl-carrier protein]	M00398	clpnC160_c	cardiolipin (tetrahexadecanoyl, n-C16:0)
C00463	Ind_c	Indole	C04778	N15PaDrbs56dimben_c	N1-(5-Phospho-alpha-D-ribosyl)-5,6-dimethylbenzimidazole	M00066	tiC162eACP_c	trans-isohexadec-2-enoyl-[acyl-carrier protein]	M00399	clpnC180_c	cardiolipin (tetraoctadecanoyl, n-C18:0)
C00469	Ethnol_c	Ethanol	C04807	dAmi78dih4hyd6dipt_c	2-Amino-7,8-dihydro-4-hydroxy-6-(diphosphooxymethyl)pteridine	M00067	iC16ACP_c	Isopalmitoyl-ACP (iso-C16:0ACP)	M00400	pgi130_c	Phosphatidylglycerol (di-isotridecanoyl, iso-C13:0)
C00472	pBen_c	p-Benzoquinone	C04823	u5Prb5ami4Nsucimi_c	1-(5'-Phosphoribosyl)-5-amino-4-(N-succinocarboxamide)-imidazole	M00068	t3oiC18ACP_c	3-Oxoisooctadecanoyl-[acyl-carrier protein]	M00401	pgai130_c	Phosphatidylglycerol (dianteisotridecanoyl,ante-iso-C13:0)
C00475	Cyt_c	Cytidine	C04851	MuroylLAlaDgamGluLLysDAlaDAladipundnol_c	MurAc(oyl-L-Ala-D-gamma-Glu-L-Lys-D-Ala-D-Ala)-diphospho-undecaprenol	M00069	t3hiC18ACP_c	(R)-3-Hydroxyisooctadecanoyl-[acyl-carrier protein]	M00402	pgai171_c	Phosphatidylglycerol (dianteisoheptadecenoyl,ante-iso-C17:1)
C00483	Tyr_c	Tyramine	C04874	dAmi4hyd6Dery12-3tri78dih_c	2-Amino-4-hydroxy-6-(D-erythro-1,2,3-trihydroxypropyl)-7,8-dihydropteridine	M00070	tiC182eACP_c	trans-isooctadec-2-enoyl-[acyl-carrier protein]	M00403	pgi171_c	Phosphatidylglycerol (diiso-heptadecenoyl, iso-C17:1)
C00488	Form_c	Formamide	C04877	UDPNactoylLalanylgamDglumes26dia_c	UDP-N-acetylmuramoyl-L-alanyl-gamma-D-glutamyl-meso-2,6-diaminopimelate	M00071	iC18ACP_c	Isooctadecanoyl-ACP (iso-C18:0ACP)	M00404	pg171_c	Phosphatidylglycerol (diheptadecenoyl, C17:1)
C00492	Raf_c	Raffinose	C04882	UDPNactoylLalanylDglu6carLlysDalanylDala_c	UDP-N-acetylmuramoyl-L-alanyl-D-glutamyl-6-carboxy-L-lysyl-D-alanyl-D-alanine	M00072	ivACP_c	Isovaleryl-ACP (iso-C5:0ACP)	M00405	pg120_c	Phosphatidylglycerol (didodecanoyl, n-C12:0)
C00493	Shi_c	Shikimate	C04895	ste8Dihy3trip_c	7,8-Dihydroneopterin 3'-triphosphate	M00073	t3oiC7ACP_c	3-Oxoisoheptanoyl-[acyl-carrier protein]	M00406	pg140_c	Phosphatidylglycerol (ditetradecanoyl, n-C14:0)
C00498	ADPglu_c	ADP-glucose	C04896	c5PDrbs15phoimi4car_c	5-(5-Phospho-D-ribosylaminoformimino)-1-(5-phosphoribosyl)-imidazole-4-carboxamide	M00074	t3hiC7ACP_c	(R)-3-Hydroxyisoheptanoyl-[acyl-carrier protein]	M00407	pg160_c	Phosphatidylglycerol (dihexadecanoyl, n-C16:0)
C00499	Allte_c	Allantoate	C04916	N5PD1rib5ami15phoDrbs4imi_c	N-(5'-Phospho-D-1'-ribulosylformimino)-5-amino-1-(5''-phospho-D-ribosyl)-4-imidazolecarboxamide	M00075	tiC72eACP_c	trans-isohep-2-enoyl-[acyl-carrier protein]	M00408	pg180_c	Phosphatidylglycerol (dioctadecanoyl, n-C18:0)
C00504	Fol_c	Folate	C05125	daHxyedp_c	2-(alpha-Hydroxyethyl)thiamine diphosphate	M00076	iC7ACP_c	Isoheptanoyl-ACP (iso-C7:0ACP)	M00409	pii130_c	phosphatidylinositol (di-isotritadecanoyl, iso-C13:0)
C00507	LRha_c	L-Rhamnose	C05198	cDxade_c	5'-Deoxyadenosine	M00077	t3oiC9ACP_c	3-Oxoisononanoyl-[acyl-carrier protein]	M00410	piai130_c	phosphatidylinositol (diante-isotritadecanoyl, ante-iso-C13:0)
C00508	LRib_c	L-Ribulose	C05223	Dodecoylacp_c	Dodecanoyl-[acyl-carrier protein]	M00078	t3hiC9ACP_c	(R)-3-Hydroxyisononanoyl-[acyl-carrier protein]	M00411	piai171_c	phosphatidylinositol (diante-isoheptadecenoyl, ante-iso-C17:1)
C00513	CDPgly_c	CDP-glycerol	C05231	L3AmiCoA_c	L-3-Aminobutyryl-CoA	M00079	tiC92eACP_c	trans-isonon-2-enoyl-[acyl-carrier protein]	M00412	pii171_c	phosphatidylinositol (diiso-heptadecenoyl, iso-C17:1)
C00522	RPante_c	(R)-Pantoate	C05258	S3Hxyhdecaoyl-CoA_c	(S)-3-Hydroxyhexadecanoyl-CoA	M00080	iC9ACP_c	Isononanoyl-ACP (iso-C9:0ACP)	M00413	pi171_c	phosphatidylinositol (diheptadecenoyl, n-C17:1)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00526	Dxu_c	Deoxyuridine	C05259	tOpaloylCoA_c	3-Oxopalmitoyl-CoA	M00081	t3oiC11ACP_c	3-Oxoisoundecanoyl-[acyl-carrier protein]	M00414	pi120_c	phosphatidylinositol (didodecanoyl, n-C12:0)
C00527	GluCoA_c	Glutaryl-CoA	C05260	S3Hxytdecaoyl-CoA_c	(S)-3-Hydroxytetradecanoyl-CoA	M00082	t3hiC11ACP_c	(R)-3-Hydroxyisoundecanoyl-[acyl-carrier protein]	M00415	pi140_c	phosphatidylinositol (ditetradecanoyl, n-C14:0)
C00530	Hquin_c	Hydroquinone	C05261	tOtetdecoylCoA_c	3-Oxotetradecanoyl-CoA	M00083	tiC112eACP_c	trans-isoundec-2-enoyl-[acyl-carrier protein]	M00416	pi160_c	phosphatidylinositol (dihexadecanoyl, n-C16:0)
C00533	Nitroxid_c	Nitric oxide	C05262	S3HxydecoylCoA_c	(S)-3-Hydroxydodecanoyl-CoA	M00084	iC11ACP_c	Isoundecanoyl-ACP (iso-C11:0ACP)	M00417	pi180_c	phosphatidylinositol (dioctadecanoyl, n-C18:0)
C00536	Trip_c	Triphosphate	C05263	tOdodecoylCoA_c	3-Oxododecanoyl-CoA	M00085	t3oiC13ACP_c	3-Oxoisotridecanoyl-[acyl-carrier protein]	M00418	psi130_c	phosphatidylserine (diisotridecanoyl, iso-C13:0)
C00542	Cys_c	Cystathionine	C05264	SHxydecoylCoA_c	(S)-Hydroxydecanoyl-CoA	M00086	t3hiC13ACP_c	(R)-3-Hydroxyisotridecanoyl-[acyl-carrier protein]	M00419	psai130_c	phosphatidylserine (diante-isotritadecanoyl, ante-iso-C13:0)
C00544	Hom_c	Homogentisate	C05265	tOdecoylCoA_c	3-Oxodecanoyl-CoA	M00087	tiC132eACP_c	trans-isotridec-2-enoyl-[acyl-carrier protein]	M00420	psai171_c	phosphatidylserine (diante-isoheptadecenoyl, ante-iso-C17:1)
C00555	qAmilde_c	4-Aminobutyraldehyde	C05266	S3HxyooylCoA_c	(S)-3-Hydroxyoctanoyl-CoA	M00088	iC13ACP_c	Isotridecanoyl-ACP (iso-C13:0ACP)	M00421	psi171_c	phosphatidylserine (diisoheptadecenoyl, iso-C17:1)
C00559	Dxade_c	Deoxyadenosine	C05267	tOoctoylCoA_c	3-Oxooctanoyl-CoA	M00089	t3oiC15ACP_c	3-Oxoisopentadecanoyl-[acyl-carrier protein]	M00422	ps171_c	phosphatidylserine (diheptadecenoyl, C17:1)
C00568	qAmibzte_c	4-Aminobenzoate	C05268	SHxyoylCoA_c	(S)-Hydroxyhexanoyl-CoA	M00090	t3hiC15ACP_c	(R)-3-Hydroxyisopentadecanoyl-[acyl-carrier protein]	M00423	ps120_c	phosphatidylserine (didodecanoyl, n-C12:0)
C00576	Betaalde_c	Betaine aldehyde	C05269	tOhexoylCoA_c	3-Oxohexanoyl-CoA	M00091	tiC152eACP_c	trans-isopentadec-2-enoyl-[acyl-carrier protein]	M00424	ps140_c	phosphatidylserine (ditetradecanoyl, n-C14:0)
C00577	DGlylde_c	D-Glyceraldehyde	C05270	HexoylCoA_c	Hexanoyl-CoA	M00092	iC15ACP_c	Isopentadecanoyl-ACP (iso-C15:0ACP)	M00425	ps160_c	phosphatidylserine (dihexadecanoyl, n-C16:0)
C00582	PhenylaceCoA_c	Phenylacetyl-CoA	C05271	tranHex2enoyoyl-CoA_c	trans-Hex-2-enoyl-CoA	M00093	t3oiC17ACP_c	3-Oxoisoheptadecanoyl-[acyl-carrier protein]	M00426	ps180_c	phosphatidylserine (dioctadecanoyl, n-C18:0)
C00588	Cholp_c	Choline phosphate	C05272	traHex2enooyl-CoA_c	trans-Hexadec-2-enoyl-CoA	M00094	t3hiC17ACP_c	(R)-3-Hydroxyisoheptadecanoyl-[acyl-carrier protein]	M00427	pei130_c	phosphatidylethanolamine (diisotritadecanoyl, iso-C13:0)
C00596	dHxy24pente_c	2-Hydroxy-2,4-pentadienoate	C05273	traTde2enooyl-CoA_c	trans-Tetradec-2-enoyl-CoA	M00095	tiC172eACP_c	trans-isoheptadec-2-enoyl-[acyl-carrier protein]	M00428	peai130_c	phosphatidylethanolamine (diante-isotritadecanoyl, ante-iso-C13:0)
C00601	Phenylacelde_c	Phenylacetaldehyde	C05274	DecoylCoA_c	Decanoyl-CoA	M00096	iC17ACP_c	Isoheptadecanoyl-ACP (iso-C17:0ACP)	M00429	peai171_c	phosphatidylethanolamine (diante-isoheptadecenoyl, ante-iso-C17:1)
C00603	Ure_c	(-)-Ureidoglycolate	C05275	traDec2enooyl-CoA_c	trans-Dec-2-enoyl-CoA	M00097	d2mbACP_c	Ante-isovaleryl-ACP (ante-iso-C5:0ACP)	M00430	pei171_c	phosphatidylethanolamine (diisoheptadecenoyl, iso-C17:1)
C00615	Prothist_c	Protein histidine	C05276	traOct2enooyl-CoA_c	trans-Oct-2-enoyl-CoA	M00098	t3oaiC7ACP_c	3-Oxoante-isoheptanoyl-[acyl-carrier protein]	M00431	pe171_c	phosphatidylethanolamine (diheptadecenoyl, C17:1)
C00620	aDRibo1p_c	alpha-D-Ribose 1-phosphate	C05345	bDFr6p_c	beta-D-Fructose 6-phosphate	M00099	t3haiC7ACP_c	(R)-3-Hydroxyante-isoheptanoyl-[acyl-carrier protein]	M00432	pe120_c	phosphatidylethanolamine (didodecanoyl, n-C12:0)
C00623	snGlyc1p_c	sn-Glycerol 1-phosphate	C05359	e_c	e-	M00100	taiC72eACP_c	trans-ante-isohep-2-enoyl-[acyl-carrier protein]	M00433	pe140_c	phosphatidylethanolamine (ditetradecanoyl, n-C14:0)
C00624	NAceLglu_c	N-Acetyl-L-glutamate	C05378	bDFr16bp_c	beta-D-Fructose 1,6-bisphosphate	M00101	aiC7ACP_c	Ante-isoheptanoyl-ACP (ante-iso-C7:0ACP)	M00434	pe160_c	phosphatidylethanolamine (dihexadecanoyl, n-C16:0)
C00630	dMylprooylCoA_c	2-Methylpropanoyl-CoA	C05379	Oxal_c	Oxalosuccinate	M00102	t3oaiC9ACP_c	3-Oxoante-isononanoyl-[acyl-carrier protein]	M00435	pe180_c	phosphatidylethanolamine (dioctadecanoyl, n-C18:0)
C00631	dPDgly_c	2-Phospho-D-glycerate	C05381	tCar1hydThP_c	3-Carboxy-1-hydroxypropyl-ThPP	M00103	t3haiC9ACP_c	(R)-3-Hydroxyante-isononanoyl-[acyl-carrier protein]	M00436	pgpi130_c	Phosphatidylglycerophosphate (diisotridecanoyl, iso-C13:0)
C00632	tHxy_c	3-Hydroxyanthranilate	C05382	Sedo7p_c	Sedoheptulose 7-phosphate	M00104	taiC92eACP_c	trans-ante-isonon-2-enoyl-[acyl-carrier protein]	M00437	pgpai130_c	Phosphatidylglycerophosphate (diante-isotridecanoyl, ante-iso-C13:0)
C00636	DMann1p_c	D-Mannose 1-phosphate	C05385	DGlucte1p_c	D-Glucuronate 1-phosphate	M00105	aiC9ACP_c	Ante-isononanoyl-ACP (ante-iso-C9:0ACP)	M00438	pgpai171_c	Phosphatidylglycerophosphate (diante-isoheptadecenoyl, ante-iso-C17:1)
C00641	u2Diasngly_c	1,2-Diacyl-sn-glycerol	C05399	Meli_c	Melibiitol	M00106	t3oaiC11ACP_c	3-Oxoante-isoundecanoyl-[acyl-carrier protein]	M00439	pgpi171_c	Phosphatidylglycerophosphate (diisoheptadecenoyl, iso-C17:1)

Continuation of Table F.13

No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name	No.	Abbreviation	Name
C00645	NAceDman_c	N-Acetyl-D-mannosamine	C05400	Epi_c	Epimelibiose	M00107	t3haiC11ACP_c	(R)-3-Hydroxyante-isoundecanoyl-[acyl-carrier protein]	M00440	pgp171_c	Phosphatidylglycerophosphate (diheptadecenoyl, C17:1)
C00655	Xant5p_c	Xanthosine 5'-phosphate	C05401	tbDGalsngly_c	3-beta-D-Galactosyl-sn-glycerol	M00108	taiC112eACP_c	trans-ante-isoundec-2-enoyl-[acyl-carrier protein]	M00441	pgp120_c	Phosphatidylglycerophosphate (didodecanoyl, n-C12:0)
C00663	bDGluc1p_c	beta-D-Glucose 1-phosphate	C05402	Mel_c	Melibiose	M00109	aiC11ACP_c	Ante-isoundecanoyl-ACP (ante-iso-C11:0ACP)	M00442	pgp140_c	Phosphatidylglycerophosphate (ditetradecanoyl, n-C14:0)
C00666	LL26Diate_c	LL-2,6-Diaminoheptanedioate	C05512	Dxi_c	Deoxyinosine	M00110	t3oaiC13ACP_c	3-Oxoante-isotridecanoyl-[acyl-carrier protein]	M00443	pgp160_c	Phosphatidylglycerophosphate (dihexadecanoyl, n-C16:0)
C00668	aDGl6p_c	alpha-D-Glucose 6-phosphate	C05526	SGlunylLcys_c	S-Glutathionyl-L-cysteine	M00111	t3haiC13ACP_c	(R)-3-Hydroxyante-isotridecanoyl-[acyl-carrier protein]	M00444	pgp180_c	Phosphatidylglycerophosphate (dioctadecanoyl, n-C18:0)
C00669	gamLGluLcys_c	gamma-L-Glutamyl-L-cysteine	C05552	Bioc_c	Biocytin	M00112	taiC132eACP_c	trans-ante-isotridec-2-enoyl-[acyl-carrier protein]	M00445	iC13_c	Isotridecanoate (C13:0)
C00671	S3Myl2openad_c	(S)-3-Methyl-2-oxopentanoic acid	C05570	Erg_c	Ergothioneine	M00113	aiC13ACP_c	Ante-isotridecanoyl-ACP (ante-iso-C13:0ACP)	M00446	aiC13_c	Ante-isotridecanoate (C13:0)
C00672	dDxDribo1p_c	2-Deoxy-D-ribose 1-phosphate	C05575	Her_c	Hercynine	M00114	t3oaiC15ACP_c	3-Oxoante-isopentadecanoyl-[acyl-carrier protein]	M00447	aiC171_c	Ante-isoheptadecenoate (anteiso-C17:1)
C00673	dDxDribo5p_c	2-Deoxy-D-ribose 5-phosphate	C05653	Fry_c	Formylanthranilate	M00115	t3haiC15ACP_c	(R)-3-Hydroxyante-isopentadecanoyl-[acyl-carrier protein]	M00448	iC171_c	Isoheptadecenoate (iso-C17:1)
C00679	cDh4deoDglu_c	5-Dehydro-4-deoxy-D-glucarate	C05668	tHxynylCoA_c	3-Hydroxypropionyl-CoA	M00116	taiC152eACP_c	trans-ante-isopentadec-2-enoyl-[acyl-carrier protein]	M00449	C171_c	heptadecenoate (C17:1)
C00680	mes26Diate_c	meso-2,6-Diaminoheptanedioate	C05744	Actacp_c	Acetoacetyl-[acp]	M00117	aiC15ACP_c	Ante-isopentadecanoyl-ACP (ante-iso-C15:0ACP)	M00450	ions_c	ions pool
C00682	dHxymsald_c	2-Hydroxymuconate semi-aldehyde	C05745	Butacp_c	Butyryl-[acp]	M00118	t3oaiC17ACP_c	3-Oxoante-isoheptadecanoyl-[acyl-carrier protein]	M00450	FerrioxE_c	Ferrioxamine-E
C00683	SMylmalnylCoA_c	(S)-Methylmalonyl-CoA	C05746	tOhexoylacp_c	3-Oxohexanoyl-[acp]	M00119	t3haiC17ACP_c	(R)-3-Hydroxyante-isoheptadecanoyl-[acyl-carrier protein]	M00451	FerrioxG_c	Ferrioxamine-G
C00688	dTD4dehbLrha_c	dTDP-4-dehydro-beta-L-rhamnose	C05747	R3Hxyoylacp_c	(R)-3-Hydroxyhexanoyl-[acp]	M00120	taiC172eACP_c	trans-ante-isoheptadec-2-enoyl-[acyl-carrier protein]	M00452	FerrioxB_c	Ferrioxamine-B
C00689	aaTreh6p_c	alpha,alpha'-Trehalose 6-phosphate	C05748	tranHex2enoyoyl-acp_c	trans-Hex-2-enoyl-[acp]	M00121	aiC17ACP_c	Ante-isoheptadecanoyl-ACP (ante-iso-C17:0ACP)	M00453	hybrid_PKNP_c	Hybrid PKNP BGC product
C00691	d4635Pen_c	2,4,6/3,5-Pentahydroxycyclohexanone	C05749	Hexoylacp_c	Hexanoyl-[acp]	M00123	t3ocddec5eACP_c	3-oxo-cis-dodec-5-enoyl-[acyl-carrier protein]	M00513	pgly_c	poly-glycerol

Abbreviations

2x YT	Two times yeast tryptone medium
A	adenylation
ACP	acyl-carrier protein
AT	acyltransferase
ATPM	ATP Maintenance
BASys	Bacterial Annotation System
BCDH	Branched-chain α-keto acid dehydrogenase
BCFA	Branched-chain Fatty Acid
BGC	Biosynthetic Gene Cluster
BLAST	Basic Local Alignment Search Tool
C	condensation
CDD	Conserved Domain Database
COBRApy	COnstrain Based Reconstruction Analysis for python
Cy	Condensation/Cyclization
DC	Decarboxylase
DH	dehydratase
DM	Defined medium
DNA	Deoxyribonucleotic acid
DNA medium	Difco Nutrient Agar
DW	Dry weight
EC	Enzyme Commission
ECH	Enoyl-CoA dehydratases
ER	enoyl reductase
ESI-IT	Electrospray-ion tramp
FBA	Flux Balance Analysis
FSEOF	Flux Scanning based on Enforced Objective Flux
FVA	Flux variability analysis
GAM	Growth Associated Maintenance
GPR	Gene-Protein-Reaction
GSM	Genome Scale Model
HMGS	3-hydroxy-3-methylglutaryl-CoA synthases
HPLC MS/MS	High-Performance Liquid Chromatography tandem Mass Spectrometry
ICP	Inductively coupled plasma
ISP2	International *Streptomyces* project 2
kb	kilo bases

KEGG	Kyoto Encyclopaedia of Genes and Genomes
KR	ketoreductase
KS	ketosynthase
KS*	non-elongating ketosynthase
LB	Luria-Bertani broth
LB agar	Luria-Bertani agar
mDM	Modified defined medium
mISP2	Modified ISP2 medium
mLPM	Modified leinamycin production medium
MM	Minimal Media
MOMA	Minimal Of Metabolic Adjustment
mRNA	Messenger ribonucleic acid
MRSA	methicillin resistant *Staphylococcus aureus*
MT	methyl-transferase
NGAM	Non-Growth Associated Maintenance
NGS	Next Generation Sequencing
NRPS	non-ribosomal peptide synthetase
nt	nucleotide
ORF	Open Reading Frame
OSMAC	One strain many compounds
Ox	Oxidation
PAC	P1-phage derivative artificial chromosome
PBS	Phosphate Buffer Solution
PCP	peptidyl-carrier protein
PCR	Polymerase Chain Reaction
pHMM	profile hidden markov model
PKS	polyketide synthase
PPP	Pentose Phosphate Pathway
R3	Regeneration 3 medium
RAST	Rapid Annotation Using Subsystem Technology
RNA	Ribonucleic acid
rRNA	Ribosomal ribonucleic acid
RT	Room temperature
RT	Retention time
RT-PCR	Reverse transcription polymerase chain reaction
SAM	S-Adenosyl methionine
SCFA	Straight-chain Fatty Acid
SFM	Soya Flour Mannitol agar
SMRT	Single Molecule Real Time
SOC	Super optimal broth with catabolite repression
TAG	Triacylglycerol
tRNA	Transference ribonucleic acid
TSB	Tryptone Soya Broth
TSB/YEME	Tryptone Soy Broth and Yeast Extract Malt Extract medium (1:1)
WT	Wild-type
YEME	Yeast extract-malt extract medium

Nomenclature

ΔG	in chemistry, refers to the energy associated to a chemical reaction.
S	in systems biology, refers to a stoichiometrix matrix of size $m \times n$, that represents a metabolic network.
Z	in systems biology, refers to an objective function.
*i*VR1007	in systems biology, refers to the name of a genome scale model. The name consists of the initials of the author and the number of genes.
m/z	in chemistry, refers to the mass to charge ratio of a ion.
D	in chemistry, refers to the right orientation of a substituent of a chiral carbon.
L	in chemistry, refers to the left orientation of a substituent of a chiral carbon.
GB	in computing, refers to gigabytes, a unit of computational storage. One gigabyte is equivalent to 10^9 bytes.
GeMRA	in computing, refers to the interface coded in python generated in this study that helps in the reconstruction of GSMs and in performing simulations. GeMRA is the abbreviation of Genome model reconstruction assistant.
mzXML	in computing, it is an XML based format for MS and MS/MS metabolomics data.
R	in computing, refers to a programming language and environment for statistical analysis and graphics.
RAM	in computing, refers to a place in the computer where the operating system, applications, programs and current data are kept so they can be reached by the processor. RAM is the abbreviation of Random Access Memory.
v/v	in chemistry, refers to the percentage of volume of a compound over 100 ml of volume of solvent.
w/v	in chemistry, refers to the percentage of weight over 100 ml of volume.