Elementary Conversion Modes enable exploration of all metabolic capabilities of any organism

Tom J. Clement[1,‡], Erik B. Baalhuis[2], Bas Teusink[1], Frank J. Bruggeman[1], Robert Planqué[1,2], Daan H. de Groot[1,‡,*]

1 Systems Bioinformatics, Amsterdam Institute for Molecules, Medicines & Systems, Vrije Universiteit Amsterdam, De Boelelaan 1087, 1081 HV, Amsterdam, The Netherlands
2 Department of Mathematics, Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV, Amsterdam, The Netherlands

‡These authors contributed equally to this work.
* daanhugodegroot@gmail.com

Abstract

The metabolic capabilities of cells determine their biotechnological potential, role in ecosystems, or threat level as a pathogen. Their comprehensive experimental characterisation is generally not feasible, in particular for unculturable or extremophile microorganisms. In principle, all metabolic capabilities can be computed from an organism's annotated genome sequence using a metabolic network reconstruction. However, current computational methods cannot deal with genome-scale metabolic networks. Part of the problem is that these methods generally aim to enumerate all feasible metabolic pathways, while computation of all possible (elementally balanced) conversions between nutrients and products (elementary conversion modes, ECMs) would suffice: all ECMs of a metabolic network together span its full metabolic capability. We have implemented ECM enumeration into `ecmtool`, and illustrate its usage in several examples. This work contributes to the elucidation of the full metabolic footprint of any microorganism and provides an exhaustive list of its capabilities in ecosystems and biotechnological applications.

Keywords: Elementary Conversion Modes • Metabolic characterization • Elementary Mode Analysis • Genome-scale analysis • Stoichiometric network analysis

Introduction

Metabolism underlies most microbial behaviours. Which chemical compounds a microbe can exploit for growth, which products it can make, and at which yields, is essential information for understanding its roles in ecosystems, responses to varying conditions, and potentials for biotechnology and bioremediation. In the case of pathogens, metabolic capabilities are informative about the niches in which they thrive. The metabolic capabilities of microorganisms also inform us about the potential mutualistic and competitive interactions they may have in communities. A computational method that can enumerate all metabolic capabilities of any microbe, from its annotated genome sequence, is therefore of key importance.

In the pre-genomic era, the metabolic capabilities of microbes could only be partially assessed, using experimental physiological information and elemental balancing. Hereby, so called macrochemical equations could be calculated, which specify the stoichiometry of the conversion of nutrients into biomass (cells) and byproducts[1, 2, 3, 4]. Those methods were however severely limited: they were not exhaustive, and required physiological data and basal knowledge of metabolic pathways. This information is lacking for extremophiles or unculturable microorganisms. These methods nonetheless contributed to the improvement of biotechnological processes and to the understanding of the stoichiometry of microbial growth[3]. When several substrates can be consumed or multiple byproducts can be produced, a unique macrochemical equation can however not be derived, and the methods need to be augmented with experimental data[2]. Nowadays, in the post-genomic era, where the genome of any microorganism can be sequenced, the potential exists for comprehensive and unsupervised enumeration of metabolic capabilities of a microorganism. Yet, despite its great benefits, no such method exists.

All metabolic reactions that a microbe can catalyse, and hereby its entire metabolic network, can be determined from the metabolic-gene annotations of its genome. This network reconstruction can nowadays almost be done purely computationally (see [5] for a review of reconstruction methods). The resulting genome-scale metabolic networks, or genome-scale stoichiometric models, have been determined for thousands of species. Such a model specifies all metabolic reactions, and thereby determines all possible pathways from substrates to products, which can be captured as the set of all possible elementary flux modes (EFMs)[6, 7, 8, 9, 10]. However, the enumeration of all EFMs of a metabolic network is not yet possible, due to a severe combinatorial explosion in their number[11], so that most research has focused on calculating only a subset of the EFMs[12, 13, 14, 15]. Many EFMs share the same overall substrates to products conversion, making many of them redundant when one is only interested in enumerating the metabolic capabilities of a microorganism, given its genome information.

A more appropriate approach for the exhaustive enumeration of the metabolic capabilities of a microbe would be to focus on Elementary Conversion Modes (ECMs)[16]. These are not defined in terms of the metabolic routes through the network that underlie the conversion of substrates into products; rather, they are defined in terms of the unique end results only: the feasible stoichiometries between substrates and products. This provides a smaller set of modes because many different reaction pathways can map to the same ECM. Moreover, the number of external metabolites is generally much lower than the number of reactions, so that the ECMs are lower-dimensional objects than EFMs. For these reasons, the combinatorial explosion that prohibited the enumeration of all EFMs on a genome-scale network might disappear when enumerating ECMs. ECMs were invented by Urbanczik and Wagner and represent the minimal steady state conversions[16] (see Box 1 for details). Together, the ECMs form a minimal set that generates all steady-state substrate-to-product conversions, i.e. macrochemical equations, analogous to all EFMs being the minimal set that generates all steady-state flux distributions. Thus, ECMs focus on the influence of an organism on its environment rather than on how it achieves this.

In this work, we reformulate and extend the ECM theory. We present a Python-based program, `ecmtool`, that accepts models in the SBML-format as input[17], and gives an exhaustive and exact list of ECMs as output. `Ecmtool` provides an indirect and a direct method. The indirect method is based on the standard algorithm for this type of mathematical problems[18]. It is fast and handles medium-scale networks. The direct method uses a novel algorithm that solves many linear programs in parallel. Although this algorithm is slower on one computation core, it lends itself to massive parallelization and is scalable to much larger networks. We validate that the calculated ECMs are correct on the medium-scale `e_coli_core`-network[19], and test the scope of `ecmtool` by enumerating the ECMs of networks of various sizes and complexity. Finally, we provide a method that calculates the conversions between a user-defined set of external metabolites. This work contributes to closing the gap between any microbe's genotype and phenotype. It offers a computational toolkit for the exhaustive determination of the metabolic capacities of unculturable and extremophile microorganisms.

Box 1: Definition of ECMS

ECMs are the minimal building blocks of all net conversions by metabolic networks. We start with the stoichiometry matrix N of a metabolic network. Each column denotes a reaction, each row a metabolite. Each entry n_{ij} indicates the number of moles of metabolite i produced ($n_{ij} > 0$), or consumed ($n_{ij} < 0$), per unit reaction j. We split all reversible reactions into a forward and backward reaction, so that all reactions in N are irreversible. Some metabolites are internal to the cell and some metabolites are external. Metabolites that occur both inside and outside the cell are considered as two metabolites: one internal and the other external. We denote the index set of internal metabolites by Int. The product of the stoichiometric matrix with the vector of reaction rates, $\boldsymbol{v}$, equals the vector of the rates of change (the conversion vector), i.e. $\dot{\boldsymbol{c}} = N\boldsymbol{v}$. Metabolism is assumed to be in steady state: $\dot{\boldsymbol{c}}_i = 0$ for all $i \in Int$. The space of all steady-state conversions is given by

$$\mathcal{C} = \{\dot{\boldsymbol{c}} = N\boldsymbol{v} \mid \dot{c}_i = 0 \text{ if } i \in Int,\ v_i \geq 0\}\,. \tag{1}$$

Definition 1. *The set of* Elementary Conversion Modes (ECMs) *is the minimal set of conversions* $\{ecm_1, \ldots, ecm_K\}$ *such that each steady-state conversion can be written as a positive sum of ECMs, without the production of any external metabolite being canceled in that sum.*

We will explain the two parts of this definition using the toy network example of Figure 1. The

external metabolites A, B, BM are interconverted via internal metabolites C, D, and E. All steady-state conversions together form a geometric shape called a convex polyhedral cone (see Fig. 1**b)**). As a consequence, the steady-state conversions can be fully described by the extreme rays of this cone (blue and green in the figure). Each steady-state conversion is a positive sum of some of the extreme rays. From the extreme rays all conversions can be generated. Note here that we can only take positive sums to not violate the irreversibility constraints in Equation (1). The first part of the definition of ECMs implies that these extreme rays are Elementary Conversion Modes. The example therefore has two ECMs: A → B (blue) and 2 B → BM (green).

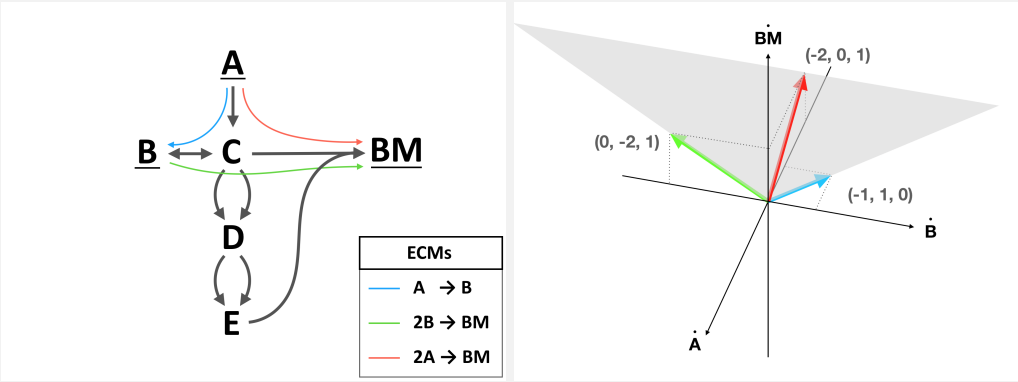

Figure 1: **a)** The Elementary Conversion Modes for a small network are shown in blue, green and red. Notice that the red ECM can be written as a positive combination of the blue and green ECM, but that this cancels the production of B. **b)** The cone of steady-state conversions is shown in gray, and is spanned by the blue and green ECM. The red ECM lies in the interior of the cone on the intersection with the $\dot{B} = 0$ -plane.

Now consider the conversion 2 A → BM (red). This conversion can be written as a positive sum of the two found ECMs: 2(A → B) + (2 B → BM)=(2 A → BM). However, in summing these ECMs, the metabolite B was canceled; since it was produced *and* consumed. Since, the ECMs are intended to capture a complete set of minimal building blocks of biologically realistic conversions, taking only the extreme rays does not suffice: it is unrealistic that a cell would simultaneously excrete and import B. The second part of the ECM-definition refers to precisely enough conversions in the set of ECMs to prevent such unrealistic cancellations. In our example, this means that the conversion 2 A → BM (red) is added. Mathematically, one can view this as finding all ECMs per orthant and taking the union of these sets (see SI for further explanation). Adding such ECMs implies that all steady-state conversions can now be written as a positive sum of ECMs, where each metabolite is either produced by all ECMs in the sum, or consumed by all ECMs in the sum. In this manner, cancellations no longer occur.

Results

ECM enumeration gives all metabolic capabilities encoded on a microorganism's genome

The number of ECMs scales better with metabolic-network size than the number of Elementary Flux Modes (Figure 2). For example, the number of elementary modes in the `e_coli_core` model[19] reduces from 100,274 EFMs to 689 ECMs. This makes visualization possible, allowing for easier exploration and analysis (Figure 3). It also indicates that many EFMs share the same overall conversion – they refer to the same metabolic capability, i.e. ECM – proving that EFM enumeration is more than what is minimally required for metabolic capability enumeration.

ECMs can be computed for metabolic networks models for which EFM-enumeration is not possible. For example, we found 273,502 possible ECMs for the pathogen *Helicobacter pylori* in a minimal medium (`iIT_341`[20] 485 metabolites and 554 reactions), while EFM enumeration ran into memory errors, most

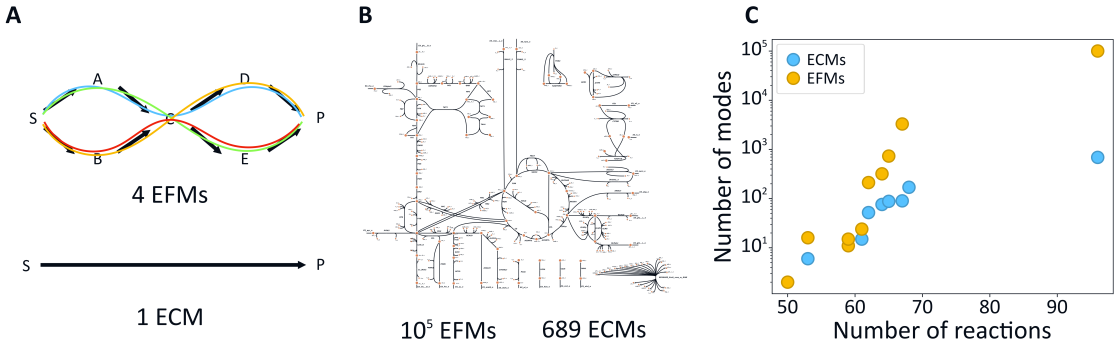

Figure 2: **The number of ECMs remains orders of magnitude lower than the number of EFMs. A)** Because many different EFMs refer to the same overall metabolic capability, the number of ECMs is much lower than the number of EFMs. **B)** EFM-vs-ECM numbers in the `e_coli_core`-network. **C)** Subnetworks of the `e_coli_core`-network were selected (see SI 11.1) to illustrate how the number of ECMs and EFMs scales with network size.

likely due to the enormous number of EFMs in this model (the full set of ECMs is available as a supplementary file). At first sight, a set of hundreds of thousands ECMs might appear hard to analyze, but the user can easily filter out a relevant subset once such a set is obtained. For example, in Supplemental Figure 7 we show all ECMs of *H. pylori* that produce biomass, without using L-alanine.

Summarizing, the enumeration of ECMs by `ecmtool` allows for the determination of all the metabolic capabilities of large metabolic networks, for which EFM enumeration is not possible. We did find that the number of ECMs in the genome-scale *E. coli* network `iJR904`[21] (761 metabolites, 1075 reactions) is still too large to be computed by `ecmtool`. Even for models of this size, however, `ecmtool` still provides useful information. Below, we describe how one can focus on the truly essential information so that even networks of this size can be analyzed.

The ECMs computed by `ecmtool` are correct and complete

We validated the results of `ecmtool` in several ways. First, we have computed the ECMs on many small models for which we could still check the correctness and completeness of the results by hand. Second, we used the `e_coli_core`-model, for which we could still use the set of EFMs enumerated by `efmtool`, to validate our results. The `Matlab`-code that we used for this validation is provided in the SI.

The correct set of ECMs should satisfy three properties: 1) each ECM must be a steady-state conversion, 2) each ECM must be an elementary vector, and 3) each steady-state conversion must be a conical combination of ECMs without metabolites being canceled in the sum.

We confirmed that all computed ECMs are steady-state conversions by checking that the net production of internal metabolites is equal to zero, and that there exists a combination of metabolic reactions that gives rise to this ECM. Then, according to the definition of ECMs given in Box 1, we proved that each ECM is elementary by showing that it cannot be written as a positive sum of the other ECMs without the production of any external metabolite being cancelled.

The third property was harder to validate, because how can we prove for *all* steady-state conversions that they can be written as a combination of ECMs? We chose to use the set of Elementary Flux Modes calculated by `efmtool`. This set spans all possible steady-state flux combinations that the model contains. For each EFM, we then calculated its overall conversion, and tried to write this conversion as a combination of ECMs. If we allowed for an error of 10^{-7}, then each conversion could be decomposed into ECMs. This error margin was necessary because the results from `efmtool` were affected by round-off errors, while the ECMs are calculated using fractions and are therefore exact.

Above, we explained and validated that `ecmtool` finds all Elementary Conversion Modes, given an *annotated* genome. The annotation is necessary for the reconstruction of the metabolic network. More strictly speaking, this minimally requires the annotation of the *metabolic* genes. Since the annotation

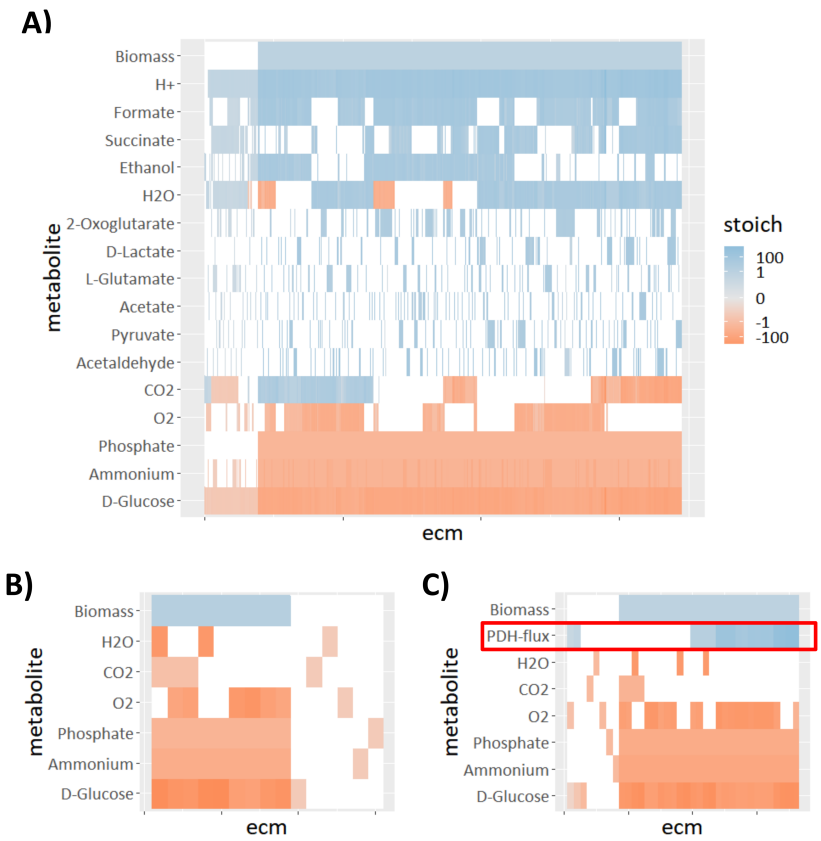

Figure 3: **The full metabolic potential of the `e_coli_core`-model. A)** The ECMs of the full model are shown as the different columns; each row corresponds to a different external metabolite. The colourscale indicates the coefficient of the metabolite in the conversion. The coefficients were log-transformed to allow for visualization of differences in both large and small coefficients (details and `R`-code can be found in the SI); small values are shown in gray, while zero values are white. Of the 689 elementary conversions, 613 lead to the production of biomass. **B)** The 15 resulting ECMs when the `hide`-method is used to focus on the uptake of metabolites and the production of biomass. This smaller set of ECMs is easier to compute and easier to explore, while the steady-state assumption is still satisfied in the whole network. So, even though the secretion of products is not reported, it has been implicitly taken into account, so that the relations between substrates shown in **A)** are still captured in **B)**. **C)** When we used the `tag`-method to report the reaction rate of pyruvate-dehydrogenase (PDH), 36 ECMs were needed to summarize all possibilities. It can be seen that the PDH-reaction is not essential for growth, but that it seems to be necessary for efficient growth on glucose since the uptake of glucose is generally lower when PDH is active.

of a genome is not always complete, we cannot guarantee that all metabolic capabilities encoded on the genome are found. We can guarantee that all conversions are found of the genome-derived metabolic network.

Box 2: Hiding and tagging enables focusing on the most important metabolic conversions

In `ecmtool`, the user can choose to compute only the stoichiometric relations between a subset of the external metabolites by 'hiding' external metabolites of minor interest. The resulting set of ECMs still gives a full summary of these relations, and complies with the steady-state

assumption on the full metabolic network. The consumption and production of the hidden metabolites still occurs, but is not reported. Therefore, ECM-enumeration with hidden metabolites can take an organism's full metabolic complexity and summarize it in only those few variables that are of interest. This `hide`-method can reduce computation time, and facilitate visual exploration of the results since the set of resulting ECMs is usually much smaller.

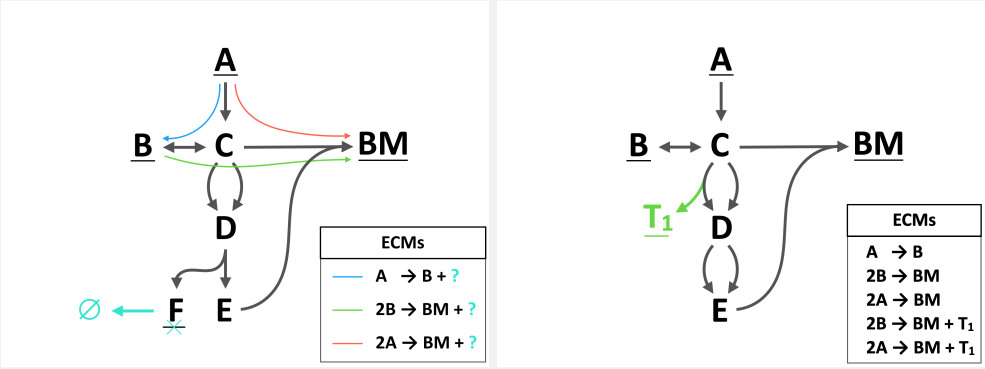

Figure 4: **A)** Information about the production of F can be ignored if F is marked as internal and a virtual reaction (cyan) is added that converts metabolite F into nothing. This strategy aids in fast computation, because the resulting set of ECMs is generally smaller. **B)** Information about the usage of a reaction can be uncovered by coupling the production of a virtual metabolite (T_1, shown in green). The coefficient of T_1 in the resulting ECMs denotes the rate of the reaction of interest.

In Figure 4**A)** we show how metabolite F can be hidden in the ECM-computation by adding a reaction that converts it into nothing (sometimes called a demand reaction). In general, a metabolite is hidden by adding a reaction that creates it from 'nothing', turns it into 'nothing', or both, depending on whether the metabolite can only be consumed, only produced, or both, respectively. Then, the metabolite is marked as an internal metabolite, so that the steady-state assumption is imposed. As a result, the added reaction can always make sure that the metabolite is not consumed nor produced, so that it will vanish from the computed conversions.

In the example of Figure 4**A)**, we obtain the conversions between non-hidden metabolites A, B and BM, ignoring the information about whether F is produced during these conversions or not. If metabolites are hidden, the computed conversions should be interpreted with care, acknowledging that the reported conversions are possibly not elementally balanced. In the example, we emphasize that we do not know whether F was produced in the conversion by adding question marks on the right side of the conversion notation. If metabolites that can be consumed by the cell are also hidden, then question marks should be placed on the left side as well.

Besides hiding metabolites of minor importance, we can keep track of reaction rates of major importance. By adding a virtual external metabolite that is produced whenever the reaction of interest is used, the usage of the reaction will be reported. This *tagging* method will show to which conversions the reaction of interest contributes, possibly providing valuable information about the essentiality of that reaction. In Figure 4**B)**, we show an example of such reaction tagging. One of two reactions from C to D is extended to produce virtual metabolite T_1, resulting in the reaction $C \rightarrow D + T_1$. Any conversion that uses the reaction of interest produces T_1, and the production coefficient is determined by the reaction rate.

In Figure 3 we illustrate the `hide`- and `tag`-methods in the `e_coli_core` model to respectively highlight the different possible combinations of growth substrates, and the necessity of the pyruvate-dehydrogenase reaction in these conversions.

Focusing on subsets of metabolites enables genome-scale calculation of conversions

Focusing on the stoichiometric relations between metabolites of major importance by hiding external metabolites of minor importance is a powerful way to scale up the size of metabolic networks that can be dealt with in `ecmtool` (see Box 2). Importantly, the steady-state assumption remains satisfied and all hidden metabolites can be produced or consumed, even though this production or consumption is not reported.

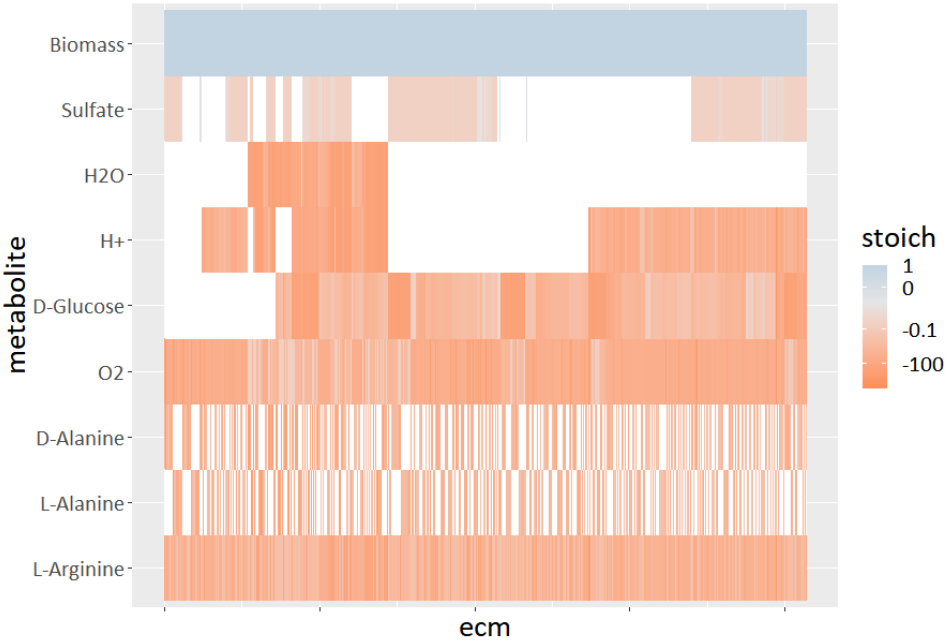

Figure 5: **Focusing on substrate uptake shows the minimal needs of *Helicobacter pylori*.** The shown ECMs were calculated for the `iIT341`-model by allowing for the uptake of all metabolites of a supposedly minimal medium proposed by the developers of the model (MinII from [20]). All output metabolites were hidden, using the method outlined in Box 2. The uptake of nine substrates is not shown here because these were equal for all ECMs, indicating that these are directly coupled to biomass formation. The colourscale indicates the log-transformed coefficients of the metabolites in the conversion (details and `R`-code can be found in the SI). The ECMs were clustered using hierarchical clustering. The block-like ordering of the ECMs shows that all different options for substrate usage can be reduced to few independent decisions.

To illustrate how the `hide`-method can help focussing on the most important metabolic capabilities of a network, we tried to find the minimal growth strategies that the pathogen *Helicobacter pylori* can employ. We took the `iIT341`-model for which we already calculated the full set of ECMs (see Figure 7), and hid all information about product secretion (Figure 5). The 8280 ECMs thus span all different nutrient uptake strategies that *H. pylori* might employ in order to grow. The results show that there is no mutual dependency between the uptake of different nutrients, except for D-Alanine, and L-Alanine, one of which should always be consumed. This independency indicates a modular design of the nutrient uptake system of *H. pylori*, which might benefit its flexibility when living in the human stomach.

If we focus only on the conversion of glucose and oxygen into biomass, we could even compute the ECMs for a genome-scale model of *E. coli*: `iJR904`[21], containing 761 metabolites, 1075 reactions (Figure 6). According to their definition, the resulting ECMs form a minimal spanning set of all feasible conversions from glucose and oxygen to biomass. This implies that the set of ECMs contains the most 'extreme' conversions. Therefore, we can use them to draw the full Pareto front between the biomass yield on glucose and on oxygen, extending a method used by Carlson and Srienc to genome-scale models(add citation)[22]. It turns out that this Pareto front is completely determined by 12 ECMs. For each of these ECMs, we can find a combination of reaction rates that gives rise to this conversion. In doing so, we

obtain twelve states of metabolism that fully determine *E. coli*'s flexibility when grown in glucose- and oxygen-limited conditions. A Flux Balance Analysis where glucose and oxygen uptake is constrained and biomass production is maximized, will always result in a combination of these metabolic states.

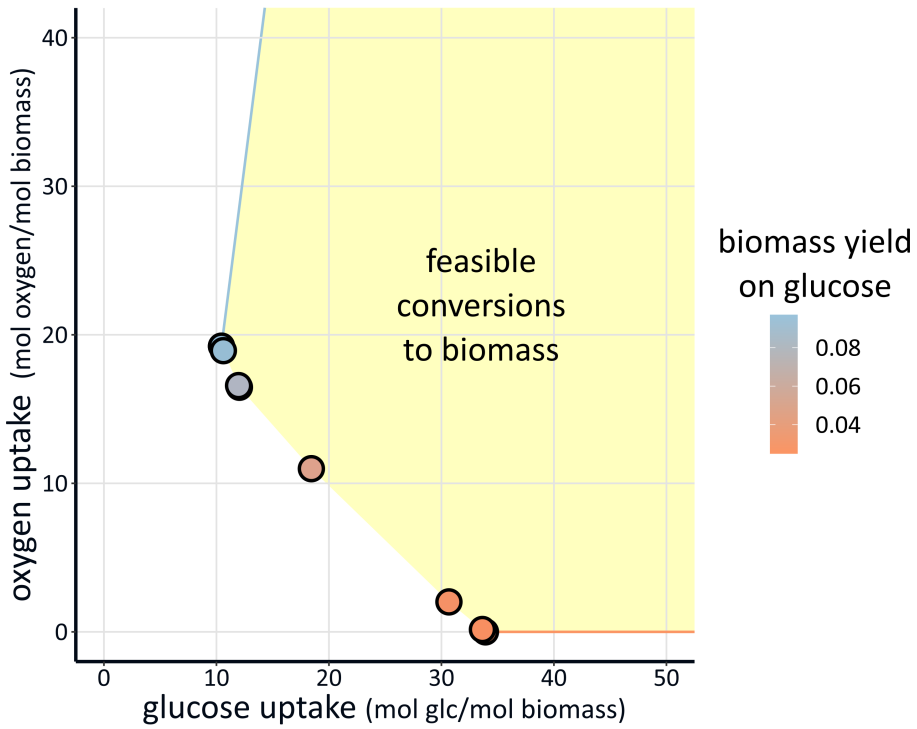

Figure 6: **Few conversions from glucose and oxygen to biomass cover *E. coli*'s full flexibility.** We calculated the ECMs for the genome-scale *E. coli*-model `iJR904`[21] by hiding all external metabolites except for glucose, oxygen and biomass. This gives 12 ECMs that span all possible biomass-yields on glucose and oxygen. The dots show the necessary glucose and oxygen uptake to produce one unit biomass, for the 10 ECMs that produce biomass. The other 2 ECMs give the most extreme conversions from glucose and oxygen to non-biomass products, consuming only glucose (red arrow), or consuming the most oxygen per glucose (blue arrow). The convex combinations of biomass-producing ECMs combined with positive multiples of the non-biomass-producing ECMs, give all feasible ways to produce one unit biomass (yellow area).

Case study: unculturable *rhizobia* could be investigated using `ecmtool`

Rhizobia are soil bacteria that can induce formation of nodule structures on plant roots, in which they differentiate into non-dividing bacteroids. Bacteroids fix atmospheric nitrogen into ammonia and make this available to the plant in exchange for carbon in the form of dicarboxylates[23]. Although a metabolic network was reconstructed, physiological information about rhizobial bacteroids is lacking, because they are difficult to isolate and extremely fragile[24]. In addition, analyzing the metabolic network with an optimization approach like FBA is unfavorable because it is unclear what the optimization objective would be. After personal correspondence, `ecmtool` was used by Schulte et al. to enumerate the metabolic capabilities of *Rhizobium leguminosarum*[25]. This aided in exposing the role of oxygen supply in the observed amino acid secretion and carbon polymer synthesis by bacteroids, and in quantitatively reproducing the carbon cost of biological nitrogen fixation.

Discussion

Our method enumerates and quantifies, for any organism for which a metabolic reconstruction has been made, all possible relations between substrate uptake, product secretion, and biomass production. This method does not rely on any optimality assumption, nor does it require experimentally obtained physiological information. It uncovers the full footprint that an organism may have on its environment. ECM enumeration stands in a long tradition of methods that pursue this goal[26]. Some of these methods attempt to find an exhaustive list of reaction pathways that a cell is capable of, for example calculating Extreme Currents[27], Elementary Flux Modes[6], or Elementary Pathways[28]. These methods all have in common that scaling to genome-scale metabolic networks is impossible because of the rapid growth of the number of pathways with network size[11]. Other methods try to view the cell as a black box and focus on what is consumed and what is produced, leading to the concepts of macrochemical equations[2, 3], direct overall reactions[1], and eventually to Elementary Conversion Modes[16]. ECM enumeration is the only method that provides a complete set of metabolic capabilities, takes reaction irreversibility into account, and scales to genome-scale networks.

Potential applications

The enumeration of ECMs facilitates the exploratory study of metabolic networks: investigation of the ECMs could spark new hypotheses and show unexpected connections. It therefore complements optimization approaches like Flux Balance Analysis[29] that are efficient at answering questions when the question is known beforehand. Even in the case that optimization approaches are more efficient, elementary mode analysis provides additional insight. For example, EFM-analysis was used to understand an adaptive growth strategy of *Lactobacillus plantarum* that was observed experimentally and predicted by FBA[30]. In this specific case, the analysis could be restricted to primary metabolism which facilitated the EFM-computation, but this restriction is often biologically unreasonable. In the future, ECMs can replace the EFMs, such that this approach can be more generally applied. Carlson and Srienc[22] used the set of Elementary Flux Modes in a relatively small *E. coli* model to investigate optimized *E. coli* growth in carbon- and oxygen-limited conditions. Using this approach, they could simplify their analysis by selecting four Elementary Flux Modes that together determined all optimal growth strategies in different glucose- and oxygen-limited conditions. In Figure 6 we show that with `ecmtool` this approach can be generalized to genome-scale models.

Most analyses of metabolic networks require physiological information that is often not available. For example, it is often required to impose constraints on exchange fluxes, to choose a reaction rate that needs to be optimized, or at least to know which metabolites can be produced[2]. This hinders the investigation of species that are insufficiently characterized and difficult to culture. Moreover, for many organisms it is doubtful whether reaction rates are optimized at all, for example for pathogens or cells that are members of a multicellular organism. ECMs do not require extensive information, solely a reconstructed metabolic network. The decisive role that ECM enumeration can play in the study of unculturable or non-optimized organisms is exemplified by the recent application of `ecmtool` to investigate the symbiotic relationship of unculturable bacteroids with plants[25].

An overview of all feasible overall reactions might furthermore be useful when studying interacting species, such as crossfeeding species, host-pathogen interactions, or multi-species communities. The possible interactions are determined by what is consumed and produced by the individual species, which is exactly the information offered by the ECMs. Indeed, knowing the capabilities of one and the incapabilities of another might lay bare dependencies on which a stable community is built.

The investigation of genome-scale metabolic networks is often obstructed by the large number of reactions and the many connections that exist between seemingly unrelated parts of the network. The replacement of selected subnetworks by their overall conversions might provide a method to reduce the size and complexity of these networks. This is for example useful when only a small part of the network is of interest, but one still needs to know how this part is connected to biomass formation. In this case, `ecmtool` could be used to give a minimal set of conversions that connects the subnetwork of interest to the biomass reaction, disregarding which reactions are used for this. The computation can then even be simplified by choosing to enumerate only the extreme rays instead of all elementary vectors of the conversion cone (see Box 1 for an explanation of the difference), which is facilitated by one of the options in `ecmtool`.

Two algorithms were implemented with complementary advantages

Using `ecmtool`, the ECMs can be computed in two different ways. The *indirect method* was suggested by Urbanczik and Wagner[16], and makes use of the Double Description method[31]. This method is fast on small- to medium-scale networks, and might therefore be preferred over the *direct method*. The method is called indirect, because it first computes a large intermediate result which is then used to compute the ECMs (this is explained in the Method section). However, the intermediate result might be much larger than the final result. Therefore, the indirect method can run into memory issues while calculating the intermediate result, even though the final result is not that large.

Our newly developed, *direct method*, performs better on larger networks. This method iteratively adds a steady-state constraint for each internal metabolite of the metabolic network. During each step, many Linear Programs must be solved which costs a relatively large amount of CPU-time, but the direct method does not suffer from intermediate results that can be much larger than the final result. Moreover, the Linear Programs that have to be performed in one step are all independent, such that this method is highly parallelisable. We have parallelised the method such that large CPU-clusters can be used. It might be that GPU-parallelisation could scale up the computations even further.

It is important to note that both methods use fractions to prohibit any rounding errors from occurring. Although this slows down many of the calculations, this is necessary to maintain the accuracy of the computed ECMs. For example, for the Double Description method it is known that round-off errors can grow to a non-negligible size [31].

Methods to scale ECM computation even further

Although ECM-computation increases the size of models for which metabolic capabilities can be charted, the ECMs of genome-scale networks with thousands of reactions can still not be computed. We hope that this last scaling step can be made in the future. Even if this step cannot be made, the `hide`-method described in Box 2 enables focusing on the most relevant set of external metabolites while the steady-state constraints are still satisfied in the whole network. In Figures 3 and 5 we illustrate on an *E. coli* core model and on a genome-scale *H. pylori*-model that this method can be used to focus on the stoichiometric couplings between user-defined external metabolites, while preserving all necessary information from the whole network. This was not possible with any other method. Moreover, when we focussed only on the relations between glucose, oxygen and biomass production, the `hide`-method allowed us to scale ECM computation to the genome-scale *E. coli* model (Figure 6).

By focusing only on the relations between inputs and outputs, ECM theory scales up the networks that can be investigated but also disregards information about which reactions are used. This can be countered somewhat by two methods. First, for each ECM a simple Linear Program can be used to find a combination of reactions that leads to this conversion. This will give one out of many ways in which the ECM can come about. Second, the method of tagging reactions, explained in Box 2, can be used to retain more information about the most relevant reaction rates in the network, while still disregarding all other reaction rates. In Figure 3**C)** we used this method to highlight the use of the pyruvate-dehydrogenase reaction in the `e_coli_core`-network.

Conclusion

In this work we presented `ecmtool`, a computational tool that calculates all overall chemical conversions that a cell is capable of. By focusing on the ratios in which external metabolites are taken up and excreted, and thereby disregarding information about the internal processes in the cell, larger metabolic networks can be explored than ever. The computation of Elementary Conversion Modes enables exploratory investigation of the metabolic capabilities of any organism based on its metabolic network alone. This is especially useful in the case of unculturable and/or non-optimized organisms, since no other methods exist that chart their full metabolic landscape. We hope that ECM calculation will in the future become a standard step after metabolic network reconstruction, so that the metabolism of all known organisms will be fully characterized.

Methods

Here, we will describe only the most important conceptual steps of the Elementary Conversion Modes computation. The method that was implemented in `ecmtool` is more elaborately described and explained in the Supplementary Information.

The minimal ingredients for computing ECMs

To start the computation of ECMs we need the following ingredients

1. a stoichiometry matrix,
2. reversibility information of all reactions,
3. information on which metabolites are external or internal,
4. information on whether external metabolites can be produced, consumed or both

Our Python-implementation can automatically extract these from an `SBML`-file (Systems Biology Markup Language [17]). In the case that it is not clear whether a reaction is reversible or not, the reaction can be assumed to be reversible. Incorrectly marking a reaction as reversible can only lead to some 'false positives': computed ECMs that are in fact not possible, but not to 'false negatives'. Marking metabolites as internal or external might in some cases be ambiguous. What we mean here is that a metabolite is internal whenever the steady state assumption should be met, so that its production and consumption by the described reactions should balance out.

Splitting external metabolites into inputs and outputs enables ECM computation by extreme ray enumeration

The ECMs, formally defined in Box 1, can be described as the elementary vectors in the space of all steady-state conversions. This space is given by:

$$\mathcal{C} = \{\dot{\boldsymbol{c}} = N\boldsymbol{v} \mid \dot{c}_i = 0 \text{ if } i \in \mathit{Int},\ v_i \geq 0\}\,, \tag{2}$$

where N is the stoichiometry matrix, and Int the index set of internal metabolites.

We first consider the case that the space of steady-state conversions is contained in one orthant, i.e., that for each dimension, i, all conversions are either nonnegative ($\dot{c}_i \geq 0$), or nonpositive ($\dot{c}_i \leq 0$). In that case, the elementary vectors coincide with the spanning rays: a well-defined minimal set of vectors with which we can generate the cone by taking conical combinations (weighted sums with positive weights). Enumerating the extreme rays of a polyhedral cone is a known mathematical problem described for example by Fukuda[18].

The space $\mathcal{C}$ is generally not contained in one orthant, because some external metabolites can be used as an input ($\dot{c}_i < 0$) in some conversions, and as an output ($\dot{c}_i > 0$) in other conversions. Fortunately, we can extend our network slightly to make a new $\mathcal{C}$ that is contained in one orthant. Let A_{ex} be an external metabolite that is both an input and an output. We connect A_{ex} to two virtual metabolites: $A_{\text{ex,in}}$ and $A_{\text{ex,out}}$. These virtual metabolites are connected to A_{ex} by two irreversible reactions $A_{\text{ex,in}} \rightarrow A_{\text{ex}}$ and $A_{\text{ex}} \rightarrow A_{\text{ex,out}}$. Finally, we mark A_{ex} itself as an internal metabolite, such that it has to be kept in steady-state. As a consequence, conversions in which A_{ex} was produced must now produce $A_{\text{ex,out}}$ to maintain the steady-state assumption. Likewise, conversions in which A_{ex} was consumed must now consume $A_{\text{ex,in}}$. As such, all information about A_{ex} is stored in the production of $A_{\text{ex,out}}$ and the consumption of $A_{\text{ex,in}}$, while these new external metabolites can only be produced or consumed. Therefore, the new space of steady-state conversions is contained in a single orthant, so that we can proceed the ECM-computation by enumerating the spanning rays of this space. After the calculation we can then undo the splitting of metabolites so that we obtain the full set of ECMs (we prove this in the Supplemental Information).

Finding the ECMs is finding a generator representation of $\mathcal{C}$

The space of steady-state conversions $\mathcal{C}$ is a so-called pointed polyhedral cone. Such a cone can be described in two ways: with an *inequality representation* or with a *generator representation*.

The inequality representation is a set of vectors $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_M\}$ that give the bounds that constrain the cone. All elements in the cone: $\dot{\boldsymbol{c}} \in \mathcal{C}$, must satisfy $\boldsymbol{a}_i \cdot \dot{\boldsymbol{c}} \geq 0$ for all i. Or, as a matrix equation:

$$\mathcal{C} = \left\{ \dot{\boldsymbol{c}} \in \mathbb{R}^n \;\middle|\; A\dot{\boldsymbol{c}} \geq 0,\; A = \begin{bmatrix} \boldsymbol{a}_1^T \\ \vdots \\ \boldsymbol{a}_M^T \end{bmatrix} \right\} \qquad \text{(inequality representation).} \tag{3}$$

In the generator representation one gives a set of vectors, $\{\boldsymbol{r}_1, \ldots \boldsymbol{r}_K\}$, with which all elements in the cone can be generated by taking conical combinations:

$$\mathcal{C} = \left\{ \dot{\boldsymbol{c}} = R\boldsymbol{\lambda} \;\middle|\; \lambda_i \geq 0,\; R = \begin{bmatrix} \boldsymbol{r}_1 & \ldots & \boldsymbol{r}_K \end{bmatrix} \right\} \qquad \text{(generator representation).} \tag{4}$$

Computing the ECMs amounts to obtaining a minimal generator representation of $\mathcal{C}$, because the generators, $\boldsymbol{r}_i$, are then precisely the ECMs.

The main computation step: impose equality constraints on a large set of generators

Following Urbanczik et al.[16], we will start the computation with a cone that is too large, but for which we already have a generator representation. To be precise, we will start with the cone generated by the columns of the stoichiometry matrix:

$$\mathcal{C}_0 = \{\dot{\boldsymbol{c}} = N\boldsymbol{\lambda} \mid \lambda_i \geq 0\}. \tag{5}$$

This cone is the space of all conversions that can result from combinations of reactions of the metabolic network, no matter if these conversions meet the steady state requirement, or not. Therefore, this cone does contain the steady state conversion cone, $\mathcal{C}$, because it contains all possible conversions in steady-state. However, to get a good description of $\mathcal{C}$, we should still filter out all conversions out of steady state by imposing the steady state constraint. To compute the ECMs, we should therefore keep track of how our set of generators changes while we impose the steady-state equalities $\dot{c}_i = 0$ for each internal metabolite.

Concluding: we start with a set of generators of the cone $\mathcal{C}_0$, we impose the set of equalities given by $\dot{c}_i = 0$, and are then interested in the generators of the resulting cone. We have implemented two methods for this main part of the computation: an *indirect method* which was suggested in literature[18, 16], and a *direct method* which we developed ourselves.

The indirect method

As we will explain below, the indirect method twice uses the Double Description (DD) method[32, 31]. The DD-method computes a minimal set of generators from an inequality representation of a cone. This part of the computation is done using `polco`[33].

Although our actual starting point is a generator representation of $\mathcal{C}_0$, it is useful for now to imagine that we already have an inequality representation of $\mathcal{C}_0$. We will later explain how we obtain this representation. This inequality representation would be a set of vectors $\boldsymbol{h}_1, \ldots \boldsymbol{h}_M$, such that

$$\mathcal{C}_0 = \left\{ \dot{\boldsymbol{c}} \in \mathbb{R}^n \;\middle|\; H\dot{\boldsymbol{c}} \geq 0,\; H = \begin{bmatrix} \boldsymbol{h}_1^T \\ \vdots \\ \boldsymbol{h}_M^T \end{bmatrix} \right\}. \tag{6}$$

Given this representation, it is easy to impose a steady-state constraint $\dot{c}_i = 0$, by adding the elementary unit vector $\hat{e}_i = [0, \cdots, 0, 1, 0, \cdots, 0]^T$ both positively and negatively to the set of inequalities. This enforces $0 \leq \hat{e}_i \cdot \dot{\boldsymbol{c}} = \dot{c}_i$, and $0 \leq -\hat{e}_i \cdot \dot{\boldsymbol{c}} = -\dot{c}_i$, such that we have actually imposed $\dot{c}_i = 0$[1].

[1] In reality we implemented this by removing the ith column from the inequality constraint matrix H. We prove in the Supplemental Information that this is equivalent.

Comparing Equations (6) and (2), we see that by imposing these additional inequalities for all internal metabolites we go from an inequality representation for $\mathcal{C}_0$ to an inequality representation of the cone of steady-state conversions $\mathcal{C}$. From this, we can use the Double Description method to compute a minimal set of generators for this cone, yielding the ECMs.

It remains to be shown how we obtain an inequality representation of $\mathcal{C}_0$ from the generator representation we start with. For this, we use that $\mathcal{C}_0$ has a dual cone associated to it: $\mathcal{C}_0^*$, which has two important properties (see SI for more information and explanation):

1. the dual of the dual cone is again the cone: $(\mathcal{C}_0^*)^* = \mathcal{C}_0$,
2. the vectors in the generator representation of a cone form an inequality representation of the dual cone, and vice versa: $\text{gen}(\mathcal{C}_0) = \text{ineq}(\mathcal{C}_0^*)$.

Our computation starts from a generator representation of $\mathcal{C}_0$ (Equation (5)), but by property 2 this is also an inequality representation of its dual $\mathcal{C}_0^*$. By applying the Double Description method on this inequality representation, we can find a generator representation of $\mathcal{C}_0^*$. This generator representation is, again by property 2 an inequality representation of the dual of this dual cone. By property 1, we have thus obtained an inequality representation of $\mathcal{C}_0$, which was exactly what we needed. The steady-state constraints can now be imposed and the ECMs computed, as explained above.

Note that this indirect method heavily relies on the Double Description method since it is applied twice. We found that `polco`[33] functions well and is reasonably fast, but can run into memory issues when the networks for which we try to compute the ECMs get too large. We found that these memory issues were caused by the size of the inequality representation needed to describe $\mathcal{C}_0$, i.e., the issues arise in the first application of the DD-method. This therefore causes a computational limitation even though the generator representation of $\mathcal{C}$ (which we are eventually after) can be much smaller. This lack of control of the size of our intermediate results forms an important disadvantage of the indirect method. Therefore, we developed the direct method for the computation of ECMs for larger networks.

The direct method

Just as the indirect method, the direct method starts with the cone $\mathcal{C}_0$ introduced in Equation (5), generated by the columns of the stoichiometry matrix N. We collect these generators in a matrix $R^{(0)}$. Then, we iteratively impose the steady-state constraints, $\dot{c}_i = 0$, for all internal metabolites i. Imposing such a steady-state constraint means that we take the intersection of the cone $\mathcal{C}_0$ with the hyperplane $\dot{c}_i = 0$. The intersection is again a cone, called $\mathcal{C}^{(1)}$, which is generated by a new set of generators that we collect in a matrix $R^{(1)}$. Proceeding with $R^{(1)}$ and imposing more steady-state constraints, we will eventually end up with a set of generators for the steady-state conversion cone $\mathcal{C}$.

One of such iterations thus starts with a set of generators of $C^{(i-1)}$, collected in $R^{(i-1)}$. Now, we distribute these generators in three groups: a plus-group, a zero-group and a minus-group, depending on if the generators have $\dot{c}_i > 0$, $\dot{c}_i = 0$, or $\dot{c}_i < 0$, respectively. The generators in the plus- and minus-groups do not satisfy the steady-state constraint, and should therefore be dropped. However, each combination of a plus-generator with a minus-generator can provide a candidate generator that does satisfy $\dot{c}_i = 0$. These candidates, combined with the generators that were already in the zero-group, must contain all generators of $\mathcal{C}^{(i)}$.

However, when we combine all generators from the plus-group with the minus-group to create new generators, we will not get a minimal set of generators. In other words, the cone $\mathcal{C}^{(i)}$ could also be generated by a smaller number of generators. This might not seem like a large problem, but the number of unnecessary generators (also called redundant generators) will exponentially grow with the number of iterations, quickly causing computational infeasibility. Therefore, we have developed an *adjacency test*. This test determines for each candidate, i.e., an appropriate combination of a plus-generator with a minus-generator, if it is redundant. It does so by checking if the candidate can be written as a combination of other generators. If so, then the candidate is redundant, and should be left out of $R^{(i)}$. This test is implemented by performing a linear optimization for each candidate. In the Supplementary Information have added figures to explain our method. There, we also elaborate on the linear optimization, and explain how we optimized it to be fast enough.

Although performing many linear optimizations is in principle a very slow process, there is an important advantage: the different optimizations can be done completely independently. Therefore, we were able to parallelise this *direct method* so that it can now be run on large computation clusters.

Network compression facilitates ECM computation on large networks

ECM theory focuses on the overall conversions between external metabolites, instead of on how these conversions come about internally. This can be used to compress the network even before we start the main computation steps described above. We have implemented several compression steps that together bring large networks back to a workable size. Some of these compression steps were suggested by Urbanczik et al.[16], and some we have added in this work. In the Supplementary Information we provide proofs and more extensive explanations.

Redundant reactions can be deleted

We can delete redundant reactions using a program called `redund` from *lrslib*[34]. This program selects reactions that can be written as a conical combination of other reactions. Because the function of these reactions can always be replaced by the combination of the other reactions, it does not add functionality to the network and can therefore be removed. We found that the application of `redund` is feasible for genome-scale metabolic networks.[2]

Reversible reactions can be used to cancel a reaction and a metabolite

Each reversible reaction can be used to cancel itself and one metabolite it connects to. Say that a reversible reaction, R_1, produces an internal metabolite A, and say that there are several other reactions producing or consuming A. We prove in the SI that we can, without changing the ECMs of the network, add or subtract reaction R_1 to these other reactions such that the production or consumption of A is cancelled. After doing this for all reactions connected to A, R_1 is the only reaction left that produces A. This implies that no reaction flux is possible through R_1 in a steady-state solution, because the production of A cannot be compensated by another reaction. Therefore, we can delete both R_1 and A from the network without affecting the ECM-results.

Dead-end metabolites and connecting reactions can be deleted

Sometimes an internal metabolite can only be produced and not consumed, or vice versa. In this case, the reaction flux through the reactions connected to this metabolite has to be zero in any steady-state solution. Therefore, we can delete the metabolite and all connecting reactions without affecting the set of ECMs.

Reactions with a unique function can be used to cancel a reaction and a metabolite

Say that we have a reaction R_1 which is the sole reaction that produces a metabolite A, but that there are several reactions that consume A. Then, again without affecting the set of ECMs, we can add R_1 to these consuming reactions such that the consumption of A is exactly cancelled. The reaction R_1 is now the only reaction left that produces A, and can therefore not be active in a steady-state solution. We can thus cancel both R_1 and A.

Cycles of k reactions can cancel $k-1$ reactions and metabolites

A cycle is a set of reactions that can be used in a certain ratio such that nothing is produced nor consumed. Say that the reactions $R_1, \ldots, R_k$ form a cycle, so that with appropriate weights λ_i we have $\lambda_1 R_1 + \cdots + \lambda_k R_k = \emptyset \rightarrow \emptyset$. In addition, say that R_1 produces an internal metabolite A. In the SI we show that we can now use a trick similar to what we used with the reversible reactions. We use $\lambda_1 R_1$

[2] When dealing with more than 10,000 reactions, `redund` runs into problems, which is the reason that we could not use the program to replace the adjacency test that we use in our direct method. The number of candidates can be much larger than 10,000.

as the forward reaction, and $\lambda_2 R_2 + \cdots + \lambda_k R_k$ as the backward reaction to cancel the production and consumption of A. After doing this, R_1 will again be the only reaction producing A, so that we can delete both R_1 and A from the network. Since we compensated for the action of R_1 in the rest of the network, we will be left with a cycle using the reactions $R_2, \ldots, R_k$, on which we can use the same trick again. As such, we can delete $k-1$ reactions and $k-1$ metabolites.

The ECM-computation was implemented in Python

We implemented the above in a publicly available Python-program called `ecmtool`. It is freely available on GitHub at `https://github.com/tjclement/ecmtool`, and can additionally be installed through the Python package manager *pip*. The direct and indirect computation method are both available within the program. A manual and some worked-out examples are available in the SI.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

Conceptualization, D.H.dG. and T.J.C.; Software, T.J.C., E.B.B. and D.H.dG.; Validation, T.J.C., E.B.B. and D.H.dG.; Formal Analysis, T.J.C., E.B.B., R.P. and D.H.dG.; Resources, F.J.B. and B.T.; Writing - Original Draft, D.H.dG.; Writing - Review & Editing, T.J.C., R.P., F.J.B., B.T., and D.H.dG.; Visualization, T.J.C. and D.H.dG; Supervision, R.P., F.J.B., B.T. and D.H.dG; Funding Acquisition, F.J.B., B.T.

Acknowledgements

We thank Brett Olivier and Leen Stougie for helpful discussions, and Carolin Schulte for showing us the scope of the potential applications of `ecmtool`. Bas Teusink and Daan de Groot were supported by NWO VICI grant 865.14.005 (https://www.nwo.nl/).

References

[1] Happel J, Sellers PH. The Characterization of Complex Systems of Chemical Reactions. Chemical Engineering Communications. 1989;83(1):221–240. doi:10.1080/00986448908940663.

[2] de Hollander JA. The use of stoichiometric relations for the description and analysis of microbial cultures. Antonie van Leeuwenhoek. 1991;60(3-4):257–273. doi:10.1007/BF00430369.

[3] Heijnen SJ. Thermodynamics of microbial growth and its implications for process design; 1994.

[4] Von Stockar U, Maskow T, Liu J, Marison IW, Patiño R. Thermodynamics of microbial growth and metabolism: An analysis of the current situation. Journal of Biotechnology. 2006;121(4):517–533. doi:10.1016/j.jbiotec.2005.08.012.

[5] Mendoza SN, Olivier BG, Molenaar D, Teusink B. A systematic assessment of current genome-scale metabolic reconstruction tools. Genome Biology. 2019;20(1):158. doi:10.1186/s13059-019-1769-1.

[6] Schuster S, Hilgetag C. On Elementary Flux Modes in biochemical reaction systems at steady state. Journal of Biological Systems. 1994;2(02):165–182. doi:10.1142/s0218339094000131.

[7] Terzer M. Large scale methods to enumerate extreme rays and elementary modes. ETH Zurich; 2009.

[8] Gagneur J, Klamt S. Computation of elementary modes: a unifying framework and the new binary approach. BMC bioinformatics. 2004;5(1):175. doi:10.1186/1471-2105-5-175.

[9] Schilling CH, Schuster S, Palsson BO, Heinrich R. Metabolic pathway analysis: Basic concepts and scientific applications in the post-genomic era. Biotechnology Progress. 1999;15(3):296–303. doi:10.1021/bp990048k.

[10] Hunt KA, Folsom JP, Taffs RL, Carlson RP. Complete enumeration of Elementary Flux Modes through scalable demand-based subnetwork definition. Bioinformatics. 2014;30(11):1569–1578. doi:10.1093/bioinformatics/btu021.

[11] Klamt S, Stelling J. Combinatorial complexity of pathway analysis in metabolic networks. Molecular biology reports. 2002;29(1-2):233–236. doi:10.1023/A:1020390132244.

[12] David L, Bockmayr A. Computing Elementary Flux Modes involving a set of target reactions. IEEE/ACM transactions on computational biology and bioinformatics. 2014;11(6):1099–1107. doi:10.1109/TCBB.2014.2343964.

[13] Kelk SM, Olivier BG, Stougie L, Bruggeman FJ. Optimal flux spaces of genome-scale stoichiometric models are determined by a few subnetworks. Scientific reports. 2012;2:580.

[14] Chan SHJ, Ji P. Decomposing flux distributions into Elementary Flux Modes in genome-scale metabolic networks. Bioinformatics. 2011;27(16):2256–2262. doi:10.1093/bioinformatics/btr367.

[15] Marashi SAA, David L, Bockmayr A. Analysis of metabolic subnetworks by flux cone projection. Algorithms for Molecular Biology. 2012;7(1):17. doi:10.1186/1748-7188-7-17.

[16] Urbanczik R, Wagner C. Functional stoichiometric analysis of metabolic networks. Bioinformatics. 2005;21(22):4176–4180. doi:10.1093/bioinformatics/bti674.

[17] Hucka M, Finney A, Sauro HM, Bolouri H, Doyle JC, Kitano H, et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics. 2003;19(4):524–531. doi:10.1093/bioinformatics/btg015.

[18] Fukuda K. Frequently asked questions in polyhedral computation; 2004. https://inf.ethz.ch/personal/fukudak/polyfaq/polyfaq.html.

[19] Orth J Fleming R PB. Reconstruction and use of microbial metabolic networks: the core *Escherichia coli* metabolic model as an educational guide. EcoSal Plus. 2010;.

[20] Thiele I, Vo TD, Price ND, Palsson B. Expanded metabolic reconstruction of Helicobacter pylori (iIT341 GSM/GPR): An in silico genome-scale characterization of single- and double-deletion mutants. Journal of Bacteriology. 2005;187(16):5818–5830. doi:10.1128/JB.187.16.5818-5830.2005.

[21] Reed JL, Vo TD, Schilling CH, Palsson BO. An expanded genome-scale model of *Escherichia coli* K-12 (iJR904 GSM/GPR). Genome biology. 2003;4(9):R54–R54. doi:10.1186/gb-2003-4-9-r54.

[22] Carlson R, Srienc F. Fundamental *Escherichia coli* Biochemical Pathways for Biomass and Energy Production: Identification of Reactions. Biotechnology and Bioengineering. 2004;85(1):1–19. doi:10.1002/bit.10812.

[23] Poole P, Ramachandran V, Terpolilli J. Rhizobia: from saprophytes to endosymbionts. Nature Reviews Microbiology. 2018;16(5):291–303. doi:10.1038/nrmicro.2017.171.

[24] Lodwig E, Poole P. Metabolism of Rhizobium Bacteroids. Critical Reviews in Plant Sciences. 2003;22(1):37–78. doi:10.1080/713610850.

[25] Schulte CCM, Wheatley RM, Terpolilli JJ, Saalbach G, De Groot DH, Papachristodoulou A, et al. How legumes control nitrogen fixation by root nodule bacteria; 2020.

[26] Trinh CT, Wlaschin A, Srienc F. Elementary mode analysis: A useful metabolic pathway analysis tool for characterizing cellular metabolism; 2009.

[27] Clarke BL. Stoichiometric network analysis. Cell Biophysics. 1988;12(1):237–253. doi:10.1007/BF02918360.

[28] Schilling CH, Letscher D, Palsson BØ. Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective. Journal of Theoretical Biology. 2000;203(3):229–248. doi:https://doi.org/10.1006/jtbi.2000.1073.

[29] Orth JD, Thiele I, Palsson BØ. What is flux balance analysis? Nature biotechnology. 2010;28(3):245.

[30] Teusink B, Wiersma A, Jacobs L, Notebaart RA, Smid EJ. Understanding the adaptive growth strategy of Lactobacillus plantarumby in silico optimisation. PLoS Computational Biology. 2009;5(6). doi:10.1371/journal.pcbi.1000410.

[31] Fukuda K, Prodon A. Double description method revisited. In: Combinatorics and computer science. vol. 1120. Springer; 1996. p. 91–111.

[32] Motzkin TS. The Double Description Method, in Contributions to the Theory of Games II. Annals of Mathematics Study. 1953;28.

[33] Terzer M. Polco: A Java tool to compute extreme rays of polyhedral cones; 2009.

[34] Avis D. lrslib, version 6.0; 2015.

Supplementary figures

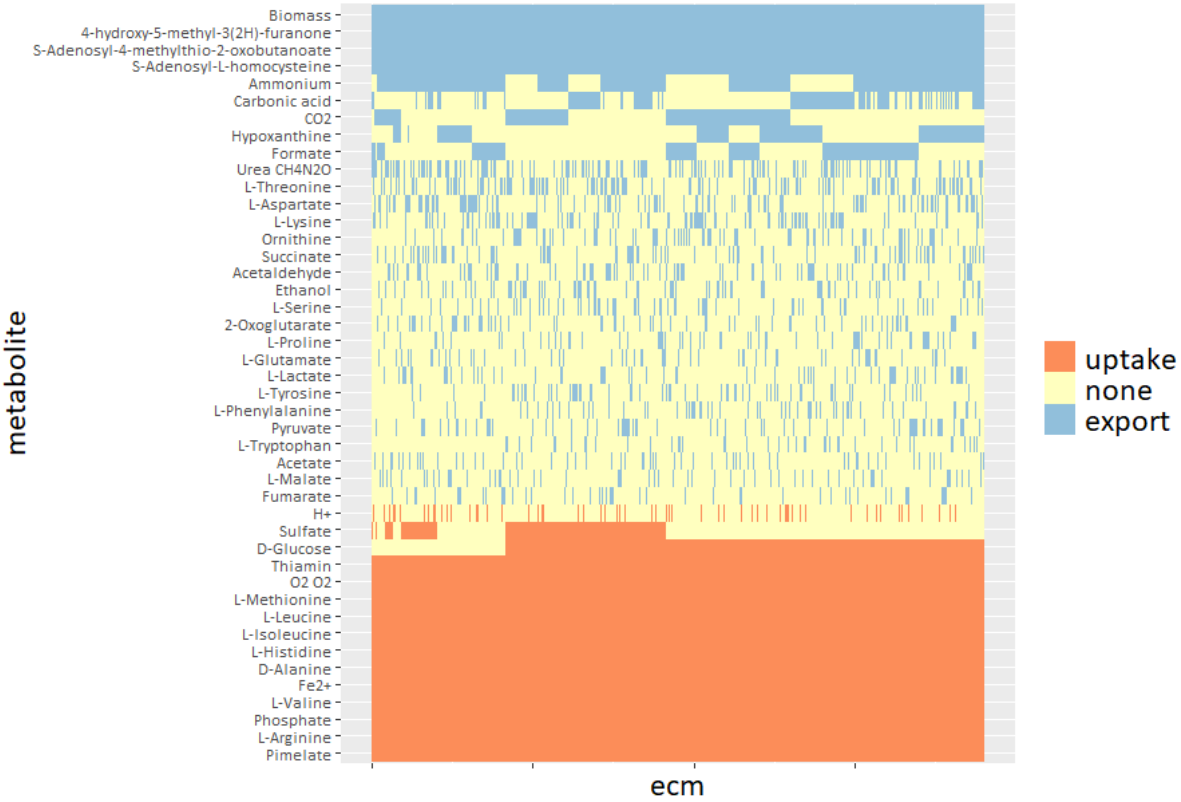

Figure 7: **All Elementary Conversion Modes for the `iIT_341`-model of *Helicobacter pylori* could be computed.** We show the ECMs that support growth on a defined medium proposed by the model developers (see minII in[20]). For visualization purposes, we only show which metabolites are consumed or produced rather than their exact stoichiometric coefficient. In addition, we assumed that L-Alanine was not taken up, since it could in all ECMs be interchanged with D-Alanine. After these simplifications, 19,023 ECMs were still unique. The full set of ECMs is available in the SI.

Supplementary Files

1. Document with detailed theoretical background for all methods used in `ecmtool`
2. User manual
3. `Matlab`-scripts used for validation of ECMs
4. `Python`-script used for creating subnetworks of the `e_coli_core`-network
5. `R`-scripts used for clustering ECMs