V21 The Double Description method: Theoretical framework behind EFM and EP

Double Description Method Revisited

Komei Fukuda[1] and Alain Prodon[2]

[1] Institute for Operations Research, ETHZ, CH-8092 Zürich, Switzerland
[2] Department of Mathematics, EPFL, CH-1015 Lausanne, Switzerland

in „Combinatorics and Computer Science Vol. 1120“ edited by Deza, Euler, Manoussakis, Springer, 1996:91

BMC Bioinformatics

BioMed Central

Research article

Open Access

Computation of elementary modes: a unifying framework and the new binary approach

Julien Gagneur[†1] and Steffen Klamt[*†2]

Address: [1]Cellzome AG, Meyerhofstr. 1, 69117 Heidelberg, Germany and [2]Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, D-39106 Magdeburg, Germany

Email: Julien Gagneur - julien.gagneur@cellzome.com; Steffen Klamt* - klamt@mpi-magdeburg.mpg.de

* Corresponding author †Equal contributors

Published: 04 November 2004

BMC Bioinformatics 2004, **5**:175 doi:10.1186/1471-2105-5-175

Received: 28 June 2004
Accepted: 04 November 2004

This article is available from: http://www.biomedcentral.com/1471-2105/5/175

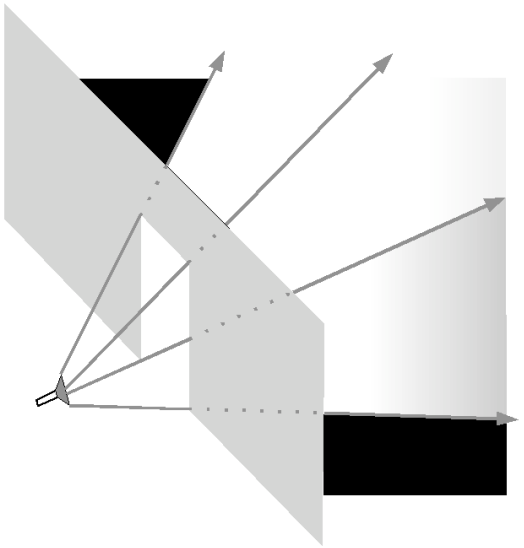
Double Description Method (1953)

All known algorithms for computing EMs are variants of the Double Description Method.

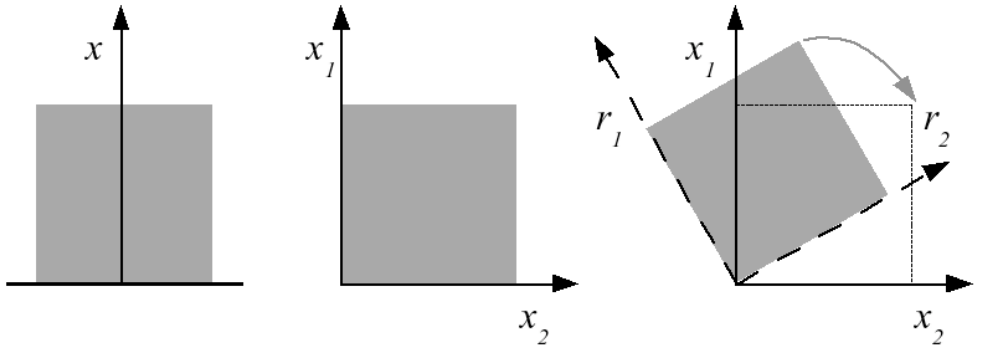
- derive simple & efficient algorithm for extreme ray enumeration, the so-called Double Description Method.

- show that it serves as a framework to the popular EM computation methods.

Analogy with Computer Graphics problem:
How can one efficiently describe the space in a dark room that is lighted by a torch shining through the open door?



Duality of Matrices



This is the duality

Left: all points above the dividing line (the shaded area) fulfill the condition $x \geq 0$.
Middle: the points in the grey area fulfill the conditions $x_1 \geq 0$ and $x_2 \geq 0$.

But how could we describe the points in the grey area on the right side in a correspondingly simple manner?
Obviously, we could define a new coordinate system (r_1, r_2) as a new set of generating vectors.
But we could also try to transform this area back into the grey area of the middle panel and use the old axes x_1 and x_2.

In 2D, this transformation can be obviously best performed by multiplying all vectors inside the grey area by a two-dimensional rotation matrix.

The Double Description Method

A pair (**A,R**) of real matrices **A** and **R** is said to be a **double description pair** or simply a ***DD* pair** if the relationship

A x ≥ **0** if and only if **x** = **R** λ for some λ ≥ **0**

holds. Clearly, for a pair *(***A,R***)* to be a *DD* pair, the column size of **A** has to equal the row size of **R**, say *d.*

For such a pair,
the set *P(***A***)* represented by **A** as $P(\mathbf{A}) = \{\mathbf{x} \in \Re^d : \mathbf{A}\mathbf{x} \geq 0\}$

is simultaneously represented by **R** as $\{\mathbf{x} \in \Re^d : \mathbf{x} = \mathbf{R}\boldsymbol{\lambda} \quad \text{for some} \quad \boldsymbol{\lambda} \geq 0\}$

A subset *P* of $\Re^d$ is called **polyhedral cone** if *P* = *P(***A***)* for some matrix **A**, and **A** is called a **representation matrix** of the polyhedral cone *P(***A***).*

Then, we say **R** is a **generating matrix** for *P*. Clearly, each column vector of a generating matrix **R** lies in the cone *P* and every vector in *P* is a nonnegative combination of some columns of **R**.

The Double Description Method

Theorem 1 (Minkowski's Theorem for Polyhedral Cones)
For any $m \times n$ real matrix **A**, there exists some $d \times m$ real matrix **R** such that (**A**,**R**) is a *DD* pair, or in other words, the cone $P(\mathbf{A})$ is generated by **R**.

The theorem states that every polyhedral cone admits a generating matrix.
The nontriviality comes from the fact that the row size of **R** is finite.
If we allow an infinite size, there is a trivial generating matrix consisting of all vectors in the cone.

Also the converse is true:

Theorem 2 (Weyl's Theorem for Polyhedral Cones)
For any $d \times n$ real matrix **R**, there exists some $m \times d$ real matrix **A** such that (**A**,**R**) is a *DD* pair, or in other words, the set generated by **R** is the cone $P(\mathbf{A})$.

The Double Description Method

Task: how does one construct a matrix **R** from a given matrix **A**, and the converse?

These two problems are computationally equivalent.
Farkas' Lemma shows that (**A**,**R**) is a *DD* pair if and only if *(*$\mathbf{R}^T$,$\mathbf{A}^T$*)* is a *DD* pair.

A more appropriate formulation of the problem is to require the minimality of **R***:*
find a matrix **R** such that no proper submatrix is generating *P(***A***)*.
A minimal set of generators is unique up to positive scaling when we assume the regularity condition that the cone is **pointed**, i.e. the origin is an extreme point of *P(***A***).*

Geometrically, the columns of a minimal generating matrix are in 1-to-1 correspondence with the **extreme rays** of **P***.*

Thus the problem is also known as the **extreme ray enumeration problem**.

No efficient (polynomial) algorithm is known for the general problem.

Double Description Method: primitive form

Suppose that the $m \times d$ matrix **A** is given and let $P(\mathbf{A}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} \geq 0\}$

(This is equivalent to the situation at the beginning of constructing EPs or EFMs: we only know **S**.)

The *DD* method is an incremental algorithm to construct a $d \times m$ matrix **R** such that (**A**,**R**) is a *DD* pair.

Let us assume for simplicity that the cone *P*(**A**) is pointed.

Let **K** be a subset of the row indices {1,2,...,*m*} of **A** and let $\mathbf{A}_\mathbf{K}$ denote the submatrix of **A** consisting of rows indexed by **K**.
Suppose we already found a generating matrix **R** for $\mathbf{A}_\mathbf{K}$, or equivalently, ($\mathbf{A}_\mathbf{K}$,**R**) is a *DD* pair. If **A** = $\mathbf{A}_\mathbf{K}$,we are done.

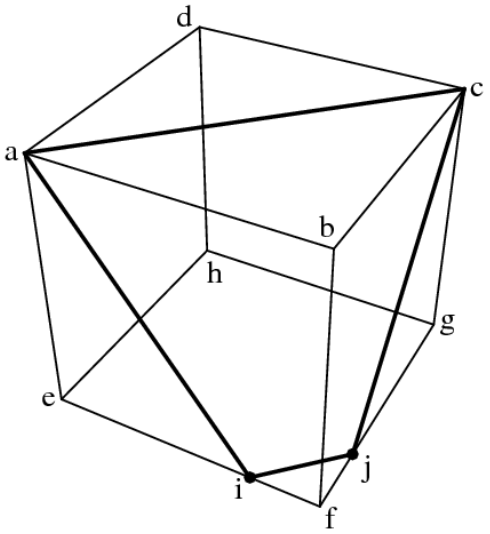
Otherwise we select any row index *i* not in **K** and try to construct a *DD* pair ($\mathbf{A}_{\mathbf{K}+\mathbf{i}}$, **R**') using the information of the *DD* pair ($\mathbf{A}_\mathbf{K}$,**R**).

Once this basic procedure is described, we have an algorithm to construct a generating matrix **R** for *P*(**A**).

Geometric version of iteration step

The procedure can be easily understood geometrically by looking at the cut-section *C* of the cone $P(\mathbf{A}_K)$ with some appropriate hyperplane *h* in $\Re^d$ which intersects with every extreme ray of $P(\mathbf{A}_K)$ at a single point.

Let us assume that the cone is pointed and thus *C* is bounded.
Having a generating matrix **R** means that all extreme rays (i.e. extreme points of the cut-section) of the cone are represented by columns of **R**.
Such a cutsection is illustrated in the Fig.



Here, *C* is the cube *abcdefgh.*

Geometric version of iteration step

The newly introduced inequality $\mathbf{A_i} \cdot \mathbf{x} \geq 0$ partitions the space $\Re^d$ into three parts:

$$H_i^+ = \{\mathbf{x} \in \Re^d : \mathbf{A_i} \cdot \mathbf{x} > 0 \}$$
$$H_i^0 = \{\mathbf{x} \in \Re^d : \mathbf{A_i} \cdot \mathbf{x} = 0 \}$$
$$H_i^- = \{\mathbf{x} \in \Re^d : \mathbf{A_i} \cdot \mathbf{x} < 0 \}$$

The intersection of H_i^0 with P and the new extreme points i and j in the cut-section C are shown in bold in the Fig.

Let J be the set of column indices of **R**. The rays $\mathbf{r_j}$ ($j \in J$) are then partitioned into three parts accordingly:

$$J^+ = \{j \in J : \mathbf{r_j} \in H_i^+ \}$$
$$J^0 = \{j \in J : \mathbf{r_j} \in H_i^0 \}$$
$$J^- = \{j \in J : \mathbf{r_j} \in H_i^- \}$$

We call the rays indexed by J^+, J^0, J^- the **positive, zero, negative** rays with respect to i, respectively.

To construct a matrix **R'** from **R**, we generate new $|J^+| \times |J^-|$ rays lying on the *ith* hyperplane H_i^0 by taking an appropriate positive combination of each positive ray $\mathbf{r_j}$ and each negative ray $\mathbf{r_{j'}}$ and by discarding all negative rays.

Geometric version of iteration step

The following lemma ensures that we have a *DD* pair ($\mathbf{A_{K+i}}$,$\mathbf{R}$‘), and provides the key procedure for the most primitive version of the *DD* method.

Lemma 3 Let *(*$\mathbf{A_K}$,$\mathbf{R}$*)* be a *DD* pair and let *i* be a row index of **A** not in **K**.
Then the pair ($\mathbf{A_{K+i}}$,$\mathbf{R}$*‘)* is a *DD* pair, where $\mathbf{R}$‘ is the $d \times |J'|$ matrix with column vectors $\mathbf{r}_j$ *(*$j \in J'$*)* defined by

$J' = J^+ \cup J^0 \cup (J^+ \times J^-)$, and

$\mathbf{r}_{jj'} = (\mathbf{A}_i \cdot \mathbf{r}_j) \cdot \mathbf{r}_{j'} - (\mathbf{A}_i \cdot \mathbf{r}_{j'}) \cdot \mathbf{r}_j$ for each $(j,j') \in J^+ \times J^-$

Proof omitted.

Finding seed *DD* pair

It is quite simple to find a *DD* pair ($\mathbf{A}_K$,$\mathbf{R}$) when $|\mathbf{K}| = 1$, which can serve as the initial *DD* pair.

Another simple (and perhaps the most efficient) way to obtain an initial *DD* form of *P* is by selecting a maximal submatrix $\mathbf{A}_K$ of $\mathbf{A}$ consisting of linearly independent rows of $\mathbf{A}$.

The vectors $\mathbf{r}_j$'s are obtained by solving the system of equations

$$\mathbf{A}_K \mathbf{R} = \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix of size $|\mathbf{K}|$, $\mathbf{R}$ is a matrix of unknown column vectors $\mathbf{r}_j$, $j \in J$.

As we have assumed rank($\mathbf{A}$) = *d,* i.e. $\mathbf{R} = \mathbf{A}_K^{-1}$ **,** the pair ($\mathbf{A}_K$,$\mathbf{R}$) is clearly a *DD* pair, since $\mathbf{A}_K \cdot \mathbf{x} \geq \mathbf{0} \leftrightarrow \mathbf{x} = \mathbf{A}_K^{-1}\lambda$, $\lambda \geq \mathbf{0}$.

Primitive algorithm for DoubleDescriptionMethod

Hence we write the *DD* method in procedural form:

```
procedure DoubleDescriptionMethod(A);
begin
    Obtain any initial DD pair (A_K, R);
    while K ≠ {1, 2, ..., m} do
    begin
        Select any index i from {1, 2, ..., m} \ K;
        Construct a DD pair (A_{K+i}, R') from (A_K, R);
            /* by using Lemma 3 */
        R := R';    K := K + i;
    end;
    Output R;
end.
```

The method given here is very primitive, and the straightforward implementation will be quite useless, because the size of *J* increases very fast and goes beyond any tractable limit.

This is because many vectors $\mathbf{r}_{jj'}$ the algorithm generates (defined in Lemma 3) are unnessary. We need to avoid generating redundant vectors.

Towards the standard implementation

Proposition 4. Let **r** be a ray of *P, G* := { **x** *:* $\mathbf{A}_{Z(\mathbf{r})} \cdot \mathbf{x} = 0$}, *F* := *G* ∩ *P* and rank($\mathbf{A}_{Z(\mathbf{r})}$) = *d* – *k*. Then

(a) rank($\mathbf{A}_{Z(\mathbf{r})\cup\{i\}}$) = *d* – *k* + 1 for all $i \notin Z(\mathbf{r})$,

(b) *F* contains *k* linearly independent rays,

(c) if $k \geq 2$ then **r** is a nonnegative combination of two distinct rays $\mathbf{r_1}$ and $\mathbf{r_2}$ with rank($\mathbf{A}_{Z(\mathbf{ri})}$) > *d* – *k, i* = 1,2.

A ray **r** is said to be **extreme** if it is not a nonnegative combination of two rays of *P* distinct from **r.**

Proposition 5. Let **r** be a ray of *P.* Then

(a) **r** is an extreme ray of *P* if and only if the rank of the matrix $\mathbf{A}_{Z(\mathbf{r})}$ is *d* – 1,

(b) **r** is a nonnegative combination of extreme rays of *P.*

Corollary 6. Let **R** be a minimal generating matrix of *P.*

Then **R** is the set of extreme rays of *P.*

Towards the standard implementation

Two distinct extreme rays **r** and **r'** of *P* are **adjacent** if the minimal face of *P* containing both contains no other extreme rays.

Proposition 7. Let **r** and **r'** be distinct rays of *P.*
Then the following statements are equivalent
(a) **r** and **r'** are adjacent extreme rays,
(b) **r** and **r'** are extreme rays and the rank of the matrix $\mathbf{A}_{Z(\mathbf{r}) \cap Z(\mathbf{r}')}$ is $d - 2$,
(c) if **r"** is a ray with $Z(\mathbf{r}") \supseteq Z(\mathbf{r}) \cap Z(\mathbf{r}')$ then either $\mathbf{r}" \cong \mathbf{r}$ or $\mathbf{r}" \cong \mathbf{r}$.

Lemma 8. Let $(\mathbf{A_K},\mathbf{R})$ be a *DD* pair such than rank($\mathbf{A_K}$) = d and let i be a row index of **A** not in K. Then the pair ($\mathbf{A_{K+i}}$, $\mathbf{R}'$) is a *DD* pair, where $\mathbf{R}'$ is the $d \times |J'|$ matrix with column vectors $\mathbf{r}_j$ ($j \in J'$) defined by

$J' = J^+ \cup J^0 \cup$ Adj

Adj = $\{(j,j') \in J^+ \times J^- : \mathbf{r}_j$ and $\mathbf{r}_{j'}$ are adjacent in $P(\mathbf{A_K})\}$ and

$\mathbf{r} = (\mathbf{A_i}\ \mathbf{r_j})\ \mathbf{r}_{j'} - (\mathbf{A_i}\mathbf{r_j})\ \mathbf{r_j}$ for each $(j,j') \in$Adj.

Furthermore, if **R** is a minimal generating matrix for $P(\mathbf{A_K})$ then $\mathbf{R}'$ is a minimal generating matrix for $P(\mathbf{A_{K+i}})$.

Algorithm for standard form of double description method

Hence we can write a straightforward variation of the *DD* method which produces a minimal generating set for P:

```
procedure : DDMethodStandard(A)
begin
    Obtain any initial DD pair (A_K, R);  such that R is minimal
    while K ≠ {1, 2, ..., m} do
    begin
        Select any index i from {1, 2, ..., m} \ K;
        Construct a DD pair (A_{K+i}, R') from (A_K, R);
            /* by using Lemma 8 */
        R := R';    K := K + i;
    end;
    Output R;
end.
```

To implement `DDMethodStandard`, we must check for each pair of extreme rays **r** and **r'** of $P(\mathbf{A}_K)$ with $\mathbf{A}_i\,\mathbf{r} > 0$ and $\mathbf{A}_i\,\mathbf{r'} < 0$ whether they are adjacent in $P(\mathbf{A}_K)$. As stated in Proposition 7, there are two ways to check adjacency, the combinatiorial and the algebraic way. While it cannot be rigorously shown which method is more efficient, in practice, the combinatorial method is always faster.

Application to central metabolism of E. coli

Redundancy removal and network compression during pre-processing results in much smaller networks.

Using a reduction of the stochiometric matrix (entries 0 and 1) allows very fast computation of even complex networks using a **binary approach**.

Table 2: Computations of elementary modes in a realistic metabolic network (central metabolism of *Escherichia coli*). Computations were performed on a typical PC with AMD Athlon XP 3000 + CPU and 1 GB RAM. Abbreviations: Form = formiate, Ac = acetate, Glc = glucose, Succ = succinate, Asp = aspartate, Glyc = glycerol, Eth = ethanol, Lac = lactate, CO2 = carbon dioxide.

	S1		S2		S3	
substrates	Glc		Glc, **Succ**, **Glyc**, **Ac**		Glc, Succ, Glyc, Ac, **Asp**	
products	Ac, Form, Eth, Lac, CO2		Ac, Form, Eth, Lac, CO2		Ac, Form, Eth, Lac, CO2, **Succ**	
#reactions (q)	106 (28 reversible)		110 (28 reversible)		112 (28 reversible)	
# metabolites (m)	89		89		89	
compressed network:						
# reactions	42 (17 reversible)		47 (17 reversible)		51 (17 reversible)	
# metabolites	25		26		28	
final number of elementary modes	27,100		507,632		2,450,787	
	binary NSA	**NSA**	**binary NSA**	**NSA**	**binary NSA**	**NSA**
computation time	0.16 min (9.63 sec)	0.54 min (32.20 sec)	51.20 min	116.77 min	1546 min (25.78 h)	not finished
back transformation	0.13 min (7.97 sec)		2.57 min		13 min	
total computation time	0.29 min (17.60 sec)	0.54 min (32.20 sec)	53.77 min	116.77 min	1559 min (25.98 h)	

Application of elementary modes
Metabolic network structure of *E.coli* determines key aspects of functionality and regulation

Table 1 **Number and distribution of elementary flux modes.**

Selection*		Glucose	Acetate	Glycerol	Succinate	Sum
-	N	27,099	598	11,332	4,249	43,279
Growth only	$N(\mu, \neq ATP)$	73.1%	58.7%	78.6%	76.3%	74.6%
ATP only	$N(\neq \mu, ATP)$	3.2%	5.0%	2.4%	2.4%	3.0%
Growth and ATP	$N(\mu, ATP)$	6.6%	2.0%	5.1%	4.2%	5.9%
No growth/ATP	$N(\neq \mu, \neq ATP)$	17.1%	34.3%	13.9%	17.1%	16.5%
Aerobic growth	$N(\mu, O_2)$	73.1%	60.7%	83.6%	80.5%	76.4%
Anaerobic growth	$N(\mu, \neq O_2)$	6.6%	0.0%	0.0%	0.0%	4.1%

*We denote the number of elementary flux modes simultaneously meeting a set of conditions, $C_1, \ldots, C_n$, by $N(C_1, \ldots, C_n)$. These conditions include, for example, the situation where cells can grow, which is abbreviated by μ. Excess energy production in the form of ATP (ATP), the substrate metabolized (S_k for the k-th substrate) and oxygen uptake (O_2) are specified accordingly. The operator ' $\neq$ ' indicates that certain fluxes must not occur. The total number of modes includes one futile cycle without substrate uptake.

Compute EFMs for central metabolism of *E.coli*.

Catabolic part: substrate uptake reactions, glycolysis, pentose phosphate pathway, TCA cycle, excretion of by-products (acetate, formate, lactate, ethanol)

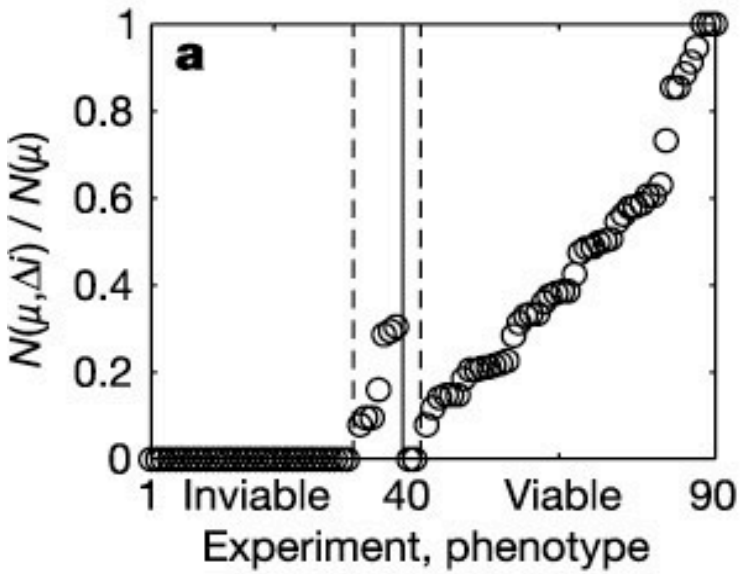
Anabolic part: conversions of precursors into building blocks like amino acids, to macromolecules, and to biomass.

Stelling et al. Nature 420, 190 (2002)

Metabolic network topology and phenotype

The total number of EFMs for given conditions is used as quantitative measure of metabolic flexibility.

a, Relative number of EFMs *N* enabling deletion mutants in gene *i* (Δ *i*) of *E. coli* to grow (abbreviated by μ) for 90 different combinations of mutation and carbon source. The solid line separates experimentally determined mutant phenotypes, namely inviability (1–40) from viability (41–90).



The # of EFMs for mutant strain allows correct prediction of growth phenotype in more than 90% of the cases.

Stelling et al. Nature 420, 190 (2002)

Relative yield

Robustness analysis

The # of EFMs qualitatively indicates whether a mutant is viable or not, but does not describe quantitatively how well a mutant grows.

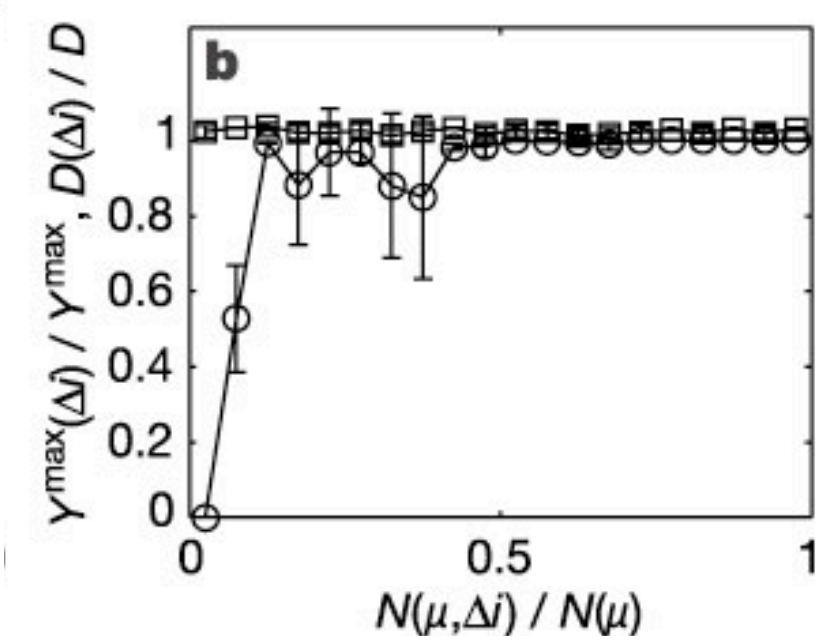
Define maximal biomass yield Y^{mass} as the optimum of:

$$Y_{i,X/S_i} = \frac{e_i^{\mu}}{e_i^{S_k}}$$

e_i is the single reaction rate (growth and substrate uptake) in EFM *i* selected for utilization of substrate S_k.

Stelling et al. Nature 420, 190 (2002)

Robustness Analysis



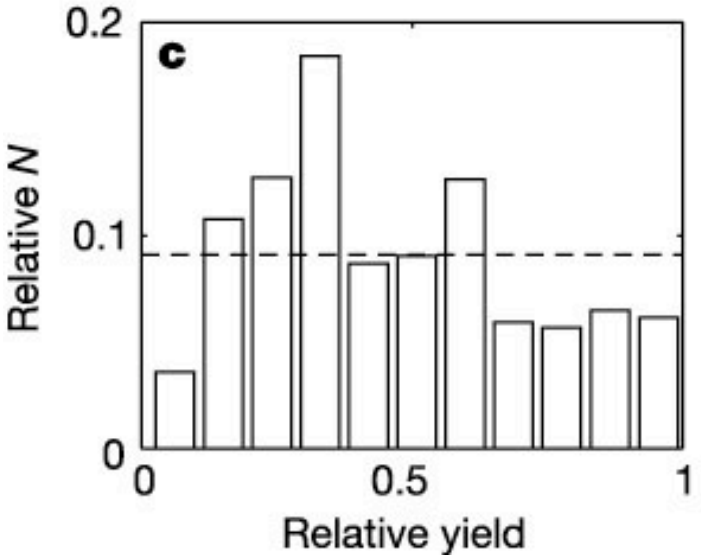
Dependency of the mutants' maximal growth yield $Y^{max}(\ i)$ (open circles) and the network diameter $D(\ i)$ (open squares) on the share of elementary modes operational in the mutants. Data were binned to reduce noise.

Stelling et al. Nature 420, 190 (2002)

Central metabolism of *E.coli* behaves in a highly robust manner because mutants with significantly reduced metabolic flexibility show a growth yield similar to wild type.

Growth-supporting elementary modes

Distribution of growth-supporting elementary modes in wild type (rather than in the mutants), that is, share of modes having a specific biomass yield (the dotted line indicates equal distribution).



Stelling et al. Nature 420, 190 (2002)

Multiple, alternative pathways exist with identical biomass yield.

Can regulation be predicted by EFM analysis?

Assume that optimization during biological evolution can be characterized by the two objectives of flexibility (associated with robustness) and of efficiency.

Flexibility means the ability to adapt to a wide range of environmental conditions, that is, to realize a maximal bandwidth of thermodynamically feasible flux distributions (maximizing # of EFMs).

Efficiency could be defined as fulfilment of cellular demands with an optimal outcome such as maximal cell growth using a minimum of constitutive elements (genes and proteins, thus minimizing # EFMs).

These 2 criteria pose contradictory challenges.
Optimal cellular regulation needs to find a trade-off.

Stelling et al. Nature 420, 190 (2002)

Can regulation be predicted by EFM analysis?

Compute control-effective fluxes for each reaction *l* by determining the efficiency of any EFM e_i by relating the system's output Ω to the substrate uptake and to the sum of all absolute fluxes.

With flux modes normalized to the total substrate uptake, **efficiencies** $\varepsilon_i(S_k, \Omega)$ for the targets for optimization Ω-growth and ATP generation, are defined as:

$$\varepsilon_i(S_k, \Omega) = \frac{e_i^{\mu}}{\sum_l |e_i^l|} \text{ and } \varepsilon_i(S_k, ATP) = \frac{e_i^{ATP}}{\sum_l |e_i^l|}$$

Control-effective fluxes $v_l(S_k)$ are obtained by averaged weighting of the product of reaction-specific fluxes and mode-specific efficiencies over all EFMs using the substrate under consideration:

$$v_l(S_k) = \frac{1}{Y^{\max}_{X/S_k}} \cdot \frac{\sum_i \varepsilon_i(S_k, \mu)|e_i^l|}{\sum_l \varepsilon_i(S_k, \mu)} + \frac{1}{Y^{\max}_{A/S_k}} \cdot \frac{\sum_i \varepsilon_i(S_k, ATP)|e_i^l|}{\sum_l \varepsilon_i(S_k, ATP)}$$

$Y^{max}_{X/Si}$ and $Y^{max}_{A/Si}$ are optimal yields of biomass production and of ATP synthesis.

Control-effective fluxes represent the importance of each reaction for efficient and flexible operation of the entire network.

Stelling et al. Nature 420, 190 (2002)

Prediction of gene expression patterns

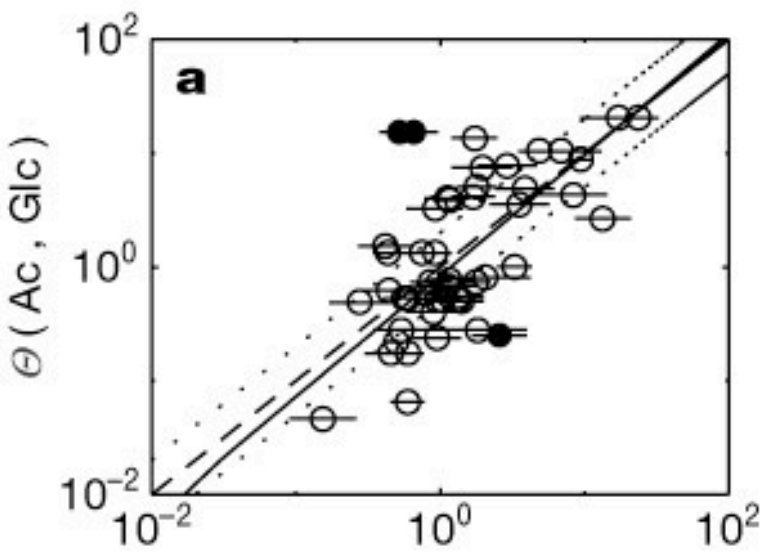
As cellular control on longer timescales is predominantly achieved by genetic regulation, the control-effective fluxes should correlate with messenger RNA levels.

Compute theoretical transcript ratios $\Theta(S_1,S_2)$ for growth on two alternative substrates S_1 and S_2 as ratios of control-effective fluxes.

Compare to exp. DNA-microarray data for E.coli growin on glucose, glycerol, and acetate.

Excellent correlation!

Stelling et al. Nature 420, 190 (2002)

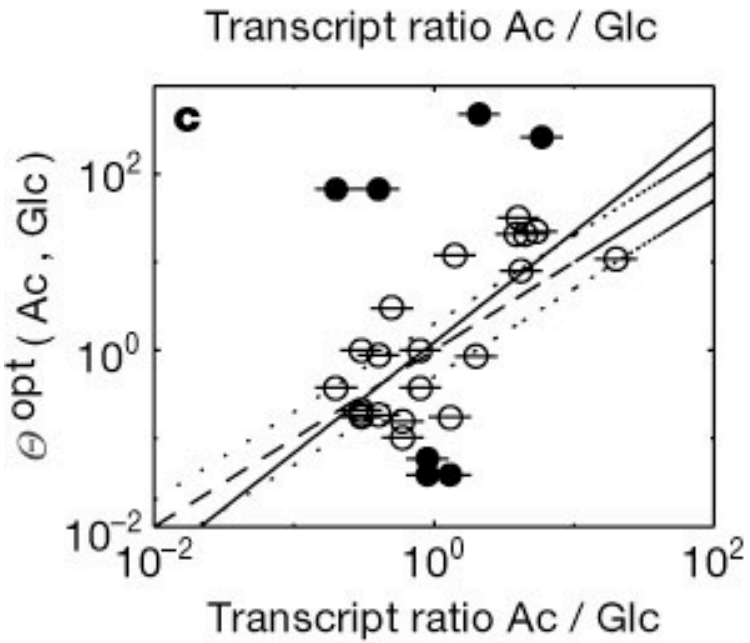


Calculated ratios between gene expression levels during exponential growth on acetate and exponential growth on glucose (filled circles indicate outliers) based on all elementary modes versus experimentally determined transcript ratios19. Lines indicate 95% confidence intervals for experimental data (horizontal lines), linear regression (solid line), perfect match (dashed line) and two-fold deviation (dotted line).

Prediction of transcript ratios

Predicted transcript ratios for acetate versus glucose for which, in contrast to **a**, only the two elementary modes with highest biomass and ATP yield (optimal modes) were considered.

This plot shows only weak correlation. This corresponds to the approach followed by Flux Balance Analysis.



Stelling et al. Nature 420, 190 (2002)

Summary of metabolic pathway analysis

Computational metabolomics: modelling constraints

Surviving (expressed) phenotypes must satisfy constraints imposed on the molecular functions of a cell, e.g. conservation of mass and energy.

Fundamental approach to understand biological systems: identify and formulate constraints.

Important constraints of cellular function:

(1) physico-chemical constraints
(2) Topological constraints
(3) Environmental constraints
(4) Regulatory constraints

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Physico-chemical constraints

These are „hard“ constraints: **Conservation** of **mass**, **energy** and **momentum**.

Contents of a cell are densely packed → **viscosity** can be 100 – 1000 times higher than that of water

Therefore, **diffusion rates** of macromolecules in cells are slower than in water.

Many molecules are confined inside the semi-permeable membrane → high osmolarity. Need to deal with **osmotic pressure** (e.g. Na^+K^+ pumps)

Reaction rates are determined by local concentrations inside cells

Enzyme-turnover numbers are generally less than 10^4 s^{-1}. **Maximal rates** are equal to the turnover-number multiplied by the enzyme concentration.

Biochemical reactions are driven by negative free-energy change in forward direction.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Topological constraints

The crowding of molecules inside cells leads to topological (3D)-constraints that affect both the form and the function of biological systems.

E.g. the ratio between the number of tRNAs and the number of ribosomes in an E.coli cell is about 10. Because there are 43 different types of tRNA, there is less than one full set of tRNAs per ribosome → it may be necessary to configure the genome so that rare codons are located close together.

E.g. at a pH of 7.6 *E.coli* typically contains only about 16 H^+ ions.
Remember that H^+ is involved in many metabolic reactions.
Therefore, during each such reaction, the pH of the cell changes!

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Environmental constraints

Environmental constraints on cells are time and condition dependent:
Nutrient availability, pH, temperature, osmolarity, availability of electron acceptors.

E.g. *Heliobacter pylori* lives in the human stomach at pH = 1
→ needs to produce NH_3 at a rate that will maintain ist immediate surrounding at a pH that is sufficiently high to allow survival.
Ammonia is made from elementary nitrogen → *H. pylori* has adapted by using amino acids instead of carbohydrates as its primary carbon source.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Regulatory constraints

Regulatory constraints are self-imposed by the organism and are subject to evolutionary change → they are no „hard“ constraints.

Regulatory constraints allow the cell to eliminate suboptimal phenotypic states and to confine itself to behaviors of increased fitness.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Mathematical formation of constraints

There are two fundamental types of constraints: balances and bounds.

Balances are constraints that are associated
with conserved quantities as energy, mass, redox potential, momentum
or with phenomena such as solvent capacity, electroneutrality and osmotic pressure.

Bounds are constraints that limit numerical ranges of individual variables and parameters such as concentrations, fluxes or kinetic constants.

Both bound and balance constraints limit the allowable functional states of reconstructed cellular metabolic networks.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

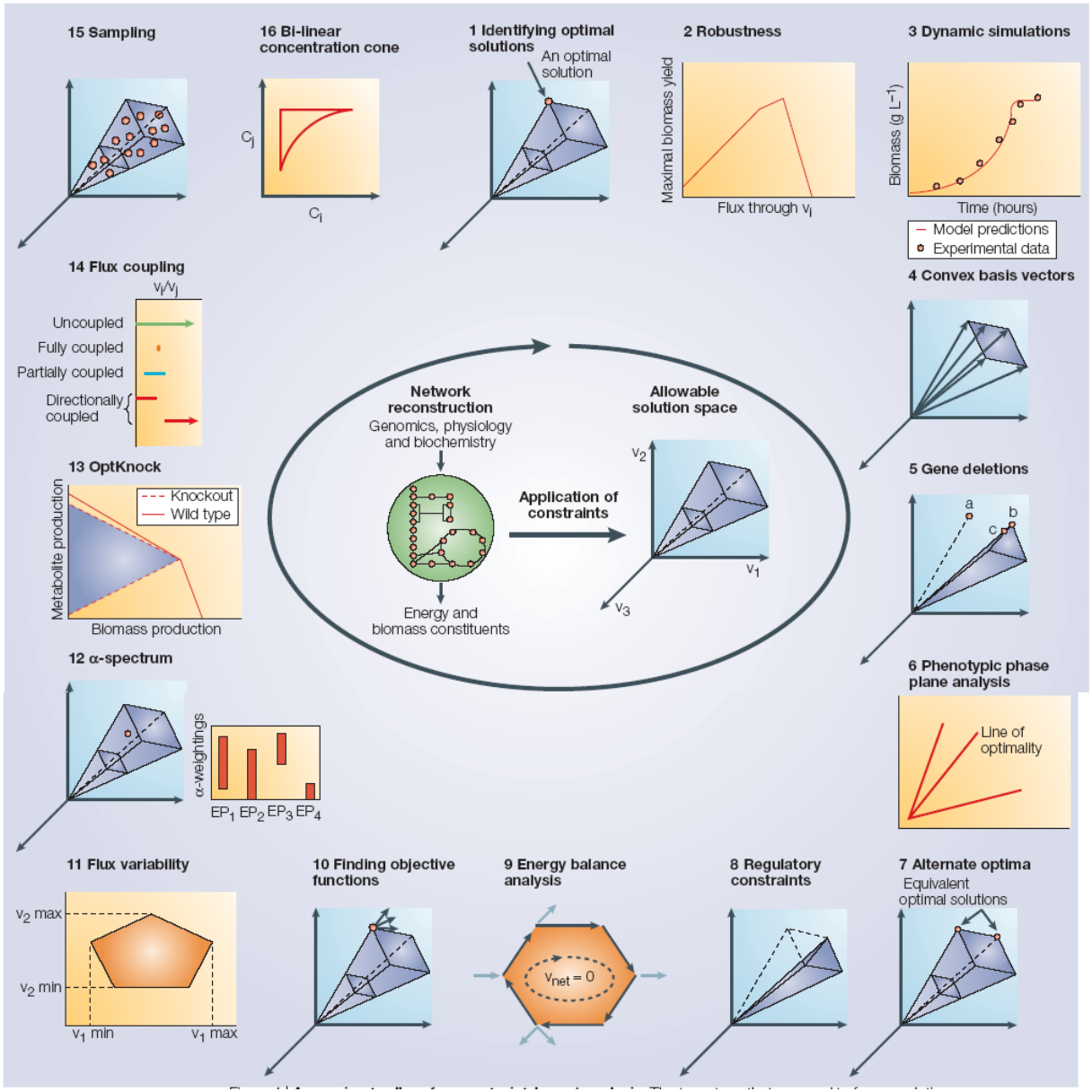
Genome-scale networks

Table 1 | **Genome-scale networks reconstructed to date**

Organism/organelle	Number of genes	Number of metabolites	Number of reactions	Year	Reference
Haemophilus influenzae	296	343	488	1999	45
Escherichia coli	660 904	436 625	720 931	2000 2003	91 19
Helicobacter pylori	291	340	388	2002	43
Saccharomyces cerevisiae	708 750	584 646	842 1,149	2003 2004	48 92
Geobacter sulfurreducens	588	541	523	2004	*
Mitochondria	N/A	230	189	2004	113

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Tools for analyzing network states



The two steps that are used to form a solution space — reconstruction and the imposition of governing constraints — are illustrated in the centre of the figure. Several methods are being developed at various laboratories to analyse the solution space.

C_i and C_j concentrations of compounds i and j;

EP, extreme pathway;

v_i and v_j fluxes through reactions i and j;

v_1 –v_3 flux through reactions 1-3;

v_{net}, net flux through loop.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Determining optimal states

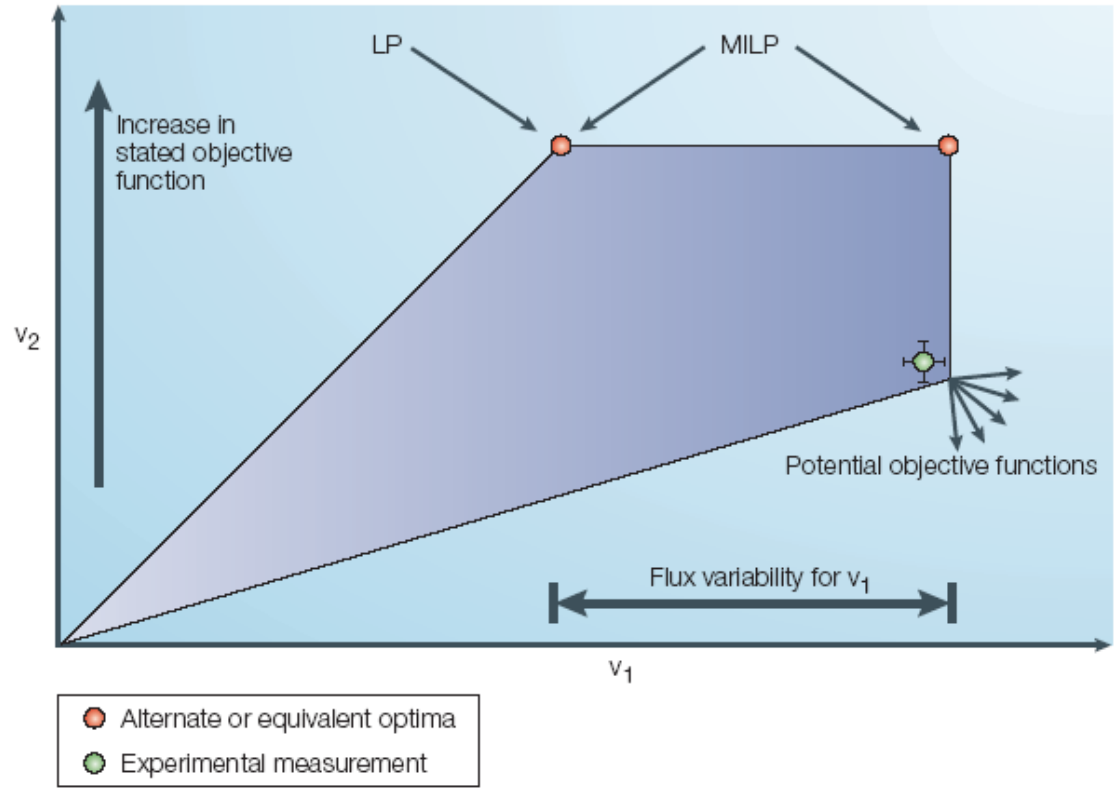


Figure 2 | **Determining optimal states.** If an objective is stated for a biochemical network, optimal solutions for the objective can be calculated. Linear programming (LP) will find one particular optimal solution, whereas mixed integer LP (MILP) can be used to find all of the basic (corner) optimal solutions. Flux variability analysis can be used to find ranges of values for all the fluxes in the set of alternate optima. In the figure, only v_1 is variable across the alternate optima. Conversely, if an objective function is not known for a biochemical network, experimental measurements can be used to identify potential objectives that would lead the cell towards that network state. v_1, flux through reaction 1; v_2, flux through reaction 2.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Flux dependencies

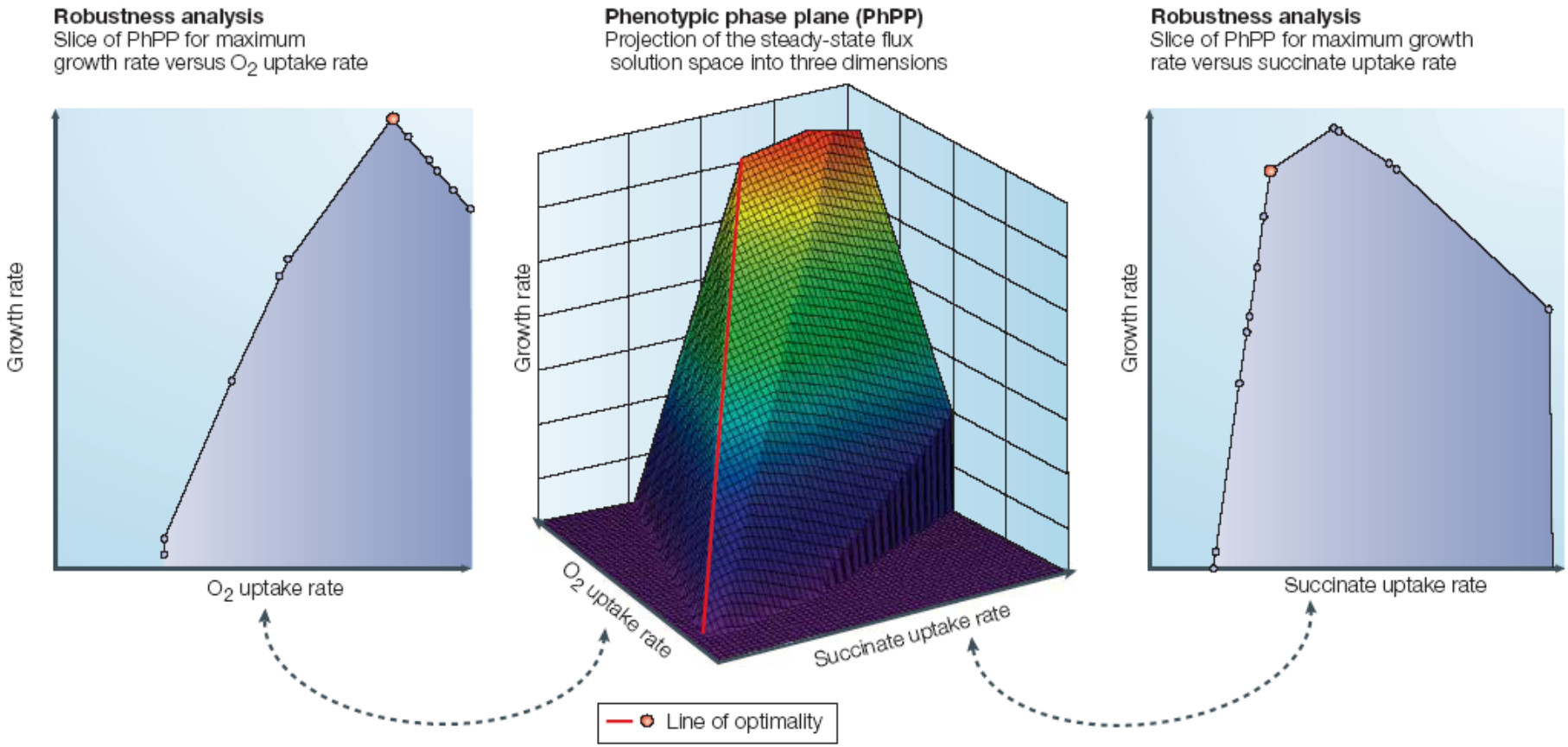


Figure 3 | **Flux dependencies.** The central figure represents a phenotypic phase plane (PhPP). It shows the maximum biomass production that is achievable at every possible combination of O_2 and succinate uptake rates. A phase plane is a projection of the solution space into two or three dimensions. The line of optimality corresponds to the conditions that are necessary for maximal biomass yield (g DW cell $mmol^{-1}$ carbon source, where DW is dry weight). Robustness analysis of the two uptake rates is shown in the two side panels. The graph on the left shows the effect on growth rate of varying O_2 uptake at a fixed succinate uptake rate. Conversely, the graph on the right shows the effect on biomass generation of varying the succinate uptake rate at a fixed O_2 uptake rate. This figure was generated using SimPheny (Genomatica, Inc.).

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Characterizing the whole solution space

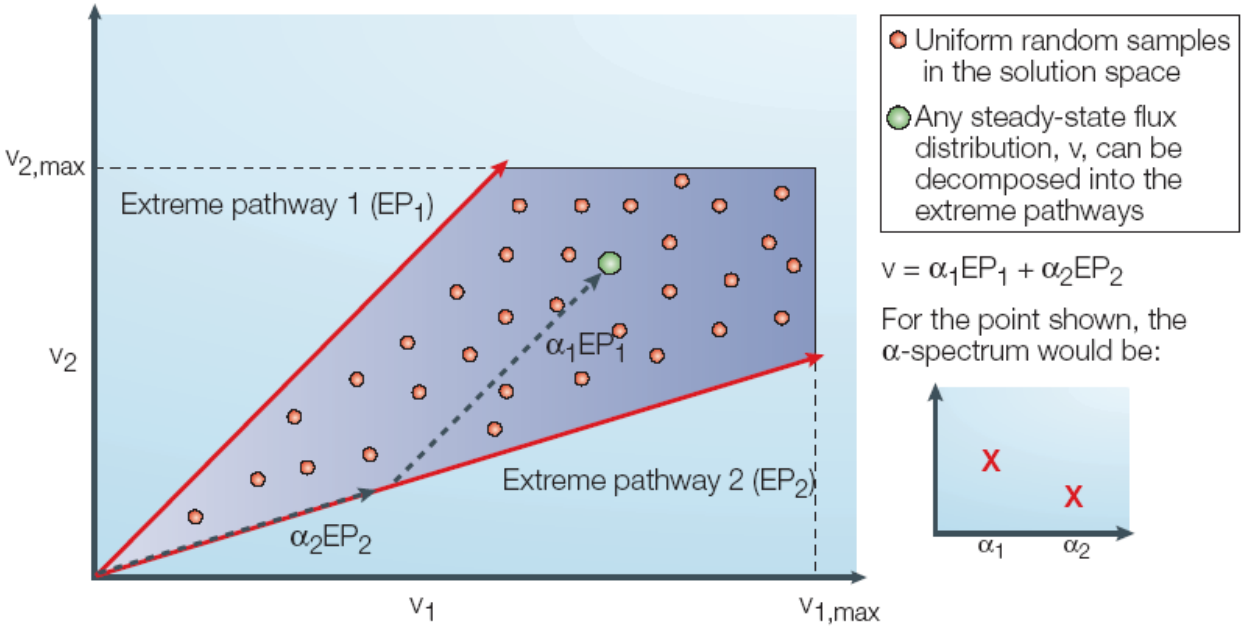


Figure 4 | **Characterizing the whole solution space.** The range of functions possible within the solution space can be characterized in two ways: first, through the definition of network-based pathways (such as the elementary modes and extreme pathways) or second, through the calculation of uniform random points within the space. The extreme pathways (EPs) are the edges of a convex space. Therefore, any point inside the space can be reached with a non-negative linear combination of the extreme pathways. In two dimensions, the decomposition of any point into two extreme pathways is unique, but in higher dimensions, the decomposition is generally non-unique. The range of possible weightings (α_1) on extreme pathways that can lead to a particular network state is called the α-spectrum. Uniform random sampling yields probability distributions for each flux based on the size and shape of the solution space and also provides a means for analysing the independence of different fluxes. v_1, flux through reaction 1; v_2, flux through reaction 2.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)

Altered solution spaces

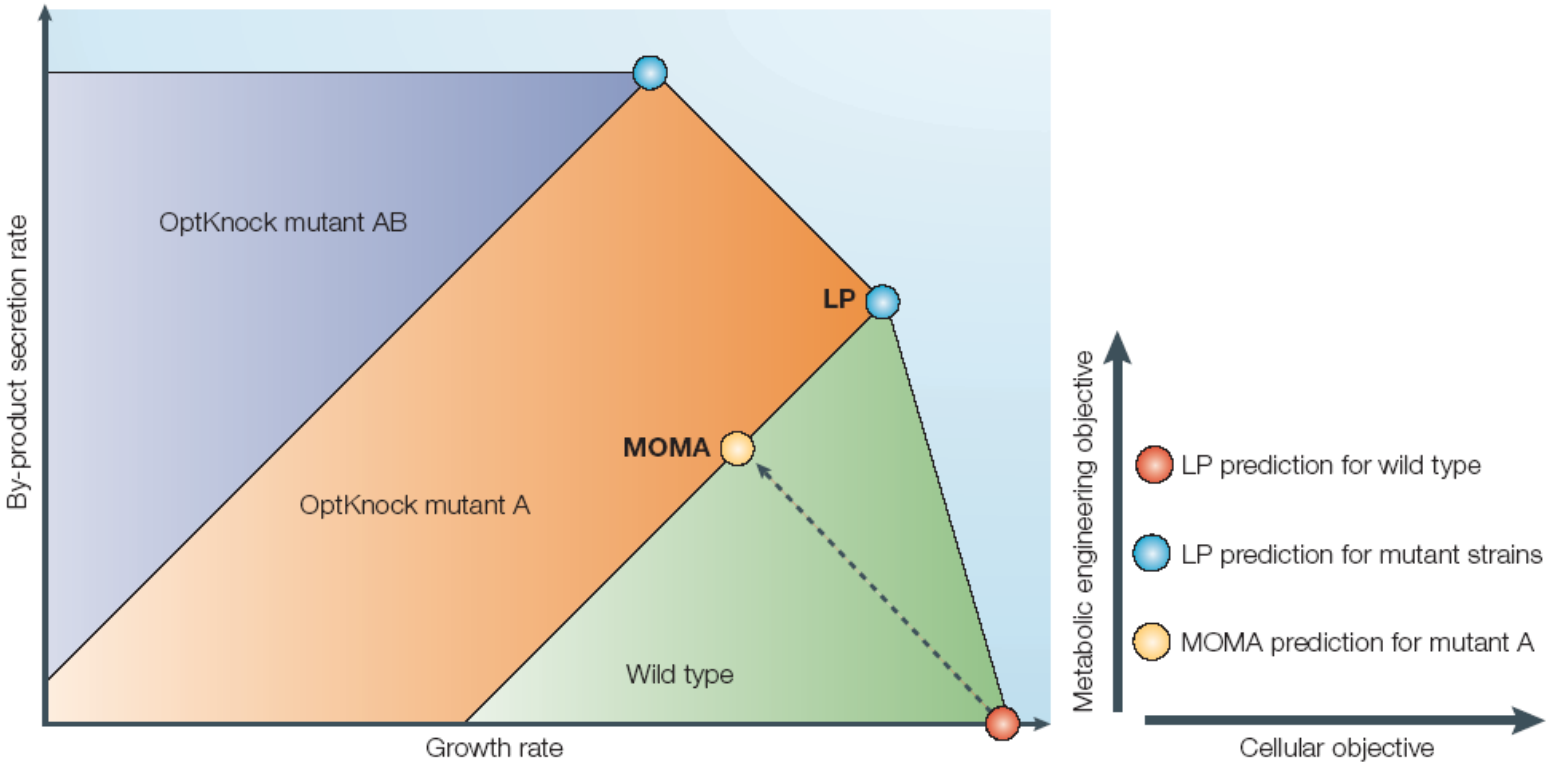


Figure 5 | **Altered solution spaces.** Solution spaces are altered by changes in the underlying biochemical network, such as occur with gene deletions. The projections of a wild-type solution space and two smaller knockout solution spaces are depicted. Optimization of growth rate (*x*-axis) in the wild-type solution space (red point) would not produce any by-product (*y*-axis), whereas optimization of growth rate in the two OptKnock mutant strains A and AB (blue points) finds solutions with by-product secretion. Minimization of metabolic adjustment (MOMA) — another method that is used for knockout predictions — assumes that, instead of being optimal for growth, the mutant will minimize the difference between its metabolic state and the metabolic state that is optimal for the wild-type strain (yellow dot). If the by-product is important, OptKnock can be used to identify knockout strains that couple optimal biomass production with by-product secretion. So, OptKnock identifies gene knockouts that require a cell to produce the desired by-product for optimal growth. In essence, the knockouts align the cell's objective with that of the metabolic engineer. Adapted with permission from REF. 35 © (2003) Wiley Interscience. LP, linear programming.

Price *et al.* Nature Rev Microbiol 2, 886 (2004)