Computation of sparsest bases in a biological context

Alexandre Temperville, PhD student
supervised by F.Lemaire and F.Boulier

Université Lille 1

JNCF 2014, Luminy

Université Lille1 Sciences et Technologies

Laboratoire d'Informatique Fondamentale de Lille

Table of Contents

Motivation

Context: modeling in biology

Use of symbolic technics for simplifying the study of biological models.

My work

What: Obtain "good" bases of conservation laws.

Why:

- each conservation law discards an equation in ODE systems,
- to prove that some concentrations are bounded.

How: Linear algebra methods.

Publication

On Defining and Computing "Good" conservation laws, F.Lemaire and A.Temperville, CMSB 2014.

Conservation laws

We consider systems of chemical reactions between species.

Definition (linear conservation law)

A linear conservation law is a linear combination of concentrations of species which is constant over time.

Example of conservation laws

$$\mathcal{S} : \begin{cases} (r_1) : A & \rightarrow B \\ (r_2) : C + D & \rightarrow \emptyset \end{cases}$$

1. $A(t) + B(t) = cst$ is a conservation law.
2. $C(t) - D(t) = cst$ is a conservation law.
3. $A(t) + B(t) + C(t) - D(t) = cst$ is a conservation law.

Computing a basis of conservation laws

Stoichiometry matrix

We can associate to a system of chemical reactions a stoichiometry matrix, with rows corresponding to species and columns to reactions.

Example

From $\mathcal{S} : \begin{cases} (r_1) : A & \rightarrow B \\ (r_2) : C + D & \rightarrow \emptyset \end{cases}$, stoichiometry matrix $M = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cc} r_1 & r_2 \end{array} \\ \begin{pmatrix} -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & -1 \end{pmatrix} \end{array}$.

Computing a basis of conservation laws

Stoichiometry matrix

We can associate to a system of chemical reactions a stoichiometry matrix, with rows corresponding to species and columns to reactions.

Example

From $\mathcal{S} : \begin{cases} (r_1) : A \rightarrow B \\ (r_2) : C + D \rightarrow \emptyset \end{cases}$, stoichiometry matrix $M = \begin{array}{c c} & \begin{array}{cc} r_1 & r_2 \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{pmatrix} -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & -1 \end{pmatrix} \end{array}$.

Any basis of $\mathrm{Ker}({}^tM)$ is a basis of conservation laws.
We will store the conservation laws by rows in a matrix B.

"Good" basis of conservation laws - sparsity

Some conservation laws seem naturally better than others.

Idea

A "good" basis of conservation laws should:

1. be **as sparse as possible**,
2. have **few negative coefficients**. (not in this talk)

Interest of sparsity

preserve sparsity: using a sparse conservation law in a system of ODEs (for reducing the number of species by 1) keeps the system **sparse**.

"Good" basis of conservation laws - positiveness

Interest of positiveness

The concentrations of species are **clearly bounded**.

Example

$A(t) + B(t) = cst \Rightarrow A(t), B(t) \in [0, cst]$.

"Good" basis of conservation laws - positiveness

Interest of positiveness

The concentrations of species are **clearly bounded**.

Example

$A(t) + B(t) = cst \Rightarrow A(t), B(t) \in [0, cst]$.

Laws with negative coefficients sometimes can't be excluded

Some systems don't have laws with only positive coefficients.

Example

$(r) : C + D \rightarrow \emptyset$ has only one conservation law: $C(t) - D(t) = cst$.

Computing sparse bases - state of art

Numerical methods computing sparse matrices

- Coleman-Pothen methods
- QR decomposition
- SVD based method
- Turnback algorithm

Computing sparse bases - state of art

Numerical methods computing sparse matrices

- Coleman-Pothen methods
- QR decomposition
- SVD based method
- Turnback algorithm

Exact methods producing (usually) sparse matrices

- RREF
- HNF
- LLL

Computing sparse bases - state of art

Numerical methods computing sparse matrices

- Coleman-Pothen methods
- QR decomposition
- SVD based method
- Turnback algorithm

Exact methods producing (usually) sparse matrices

- RREF
- HNF
- LLL

Methods computing positive conservation laws

Soliman developped a method computing P-invariants (conservation laws with non-negative coefficient with minimal support) but misses sometimes complete bases.

Computing sparse bases - state of art

Numerical methods computing sparse matrices

- Coleman-Pothen methods
- QR decomposition
- SVD based method
- Turnback algorithm

Exact methods producing (usually) sparse matrices

- RREF
- HNF
- LLL

Methods computing positive conservation laws

Soliman developped a method computing P-invariants (conservation laws with non-negative coefficient with minimal support) but misses sometimes complete bases.

These methods can sometimes give a sparsest basis.
We present a guaranteed algorithm computing a sparsest basis (published in CMSB 2014).

Table of Contents

ComputeSparsestBasis algorithm

Definition

B' is a sparsest basis if it is a basis with the fewest number of nonzeros.

ComputeSparsestBasis algorithm (CSB)

Algorithm 1: ComputeSparsestBasis(B)

Input: a basis B (stored row-wise)
Output: a sparsest basis B'

begin
  $B' \leftarrow B$;
  while *it is possible* **do**
    Replace a row of B' by a sparser row (keeping B' basis) ;

Theorem for greedy approach

The following theorem justifies the greedy approach of our algorithm.

Theorem

The basis B is not sparsest $\iff$ a row B_j can be replaced by a sparser one.

On an example

Example

Question: $B = \begin{pmatrix} 1 & 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$ is sparsest or not ? If not, how to reduce it ?

On an example

Example

Question: $B = \begin{pmatrix} 1 & 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$ is sparsest or not ? If not, how to reduce it ?

Principle of the method

Denote $\mathcal{N}(v)$ the total number of nonzero coefficients of v.
Look for a linear combination $w = \sum_{i=1}^{3} \alpha_i B_i$ and an index j s.t.:

1. $\alpha_j \neq 0$,
2. $\mathcal{N}(w) < \mathcal{N}(B_j)$.

On an example

Example

Question: $B = \begin{pmatrix} 1 & 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$ is sparsest or not ? If not, how to reduce it ?

Principle of the method

Denote $\mathcal{N}(v)$ the total number of nonzero coefficients of v.
Look for a linear combination $w = \sum_{i=1}^{3} \alpha_i B_i$ and an index j s.t.:

1. $\alpha_j \neq 0$,
2. $\mathcal{N}(w) < \mathcal{N}(B_j)$.

In theory, we can do an exhaustive search

1. Consider $w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$,
2. Compute 2^5 possible patterns with $w_i = 0$ or $w_i \neq 0$.

Binary tree

Example

$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$

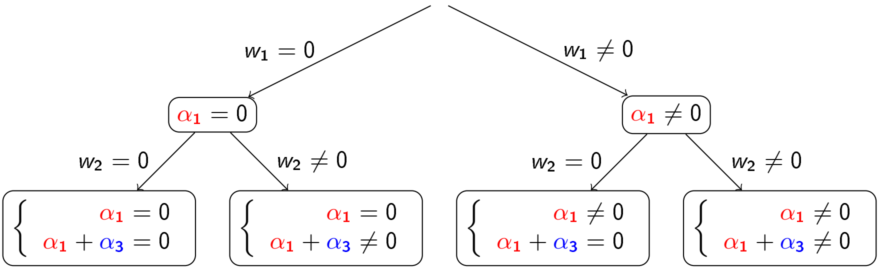
$w_1 = 0$ $\rightarrow$ $\boxed{\alpha_1 = 0}$

$w_1 \neq 0$ $\rightarrow$ $\boxed{\alpha_1 \neq 0}$

Binary tree

Example

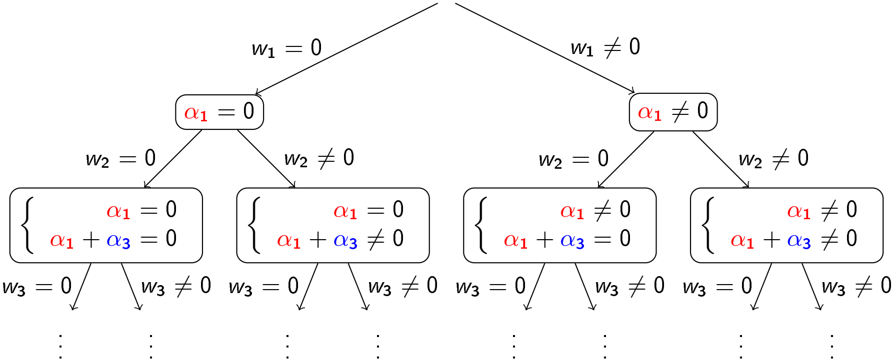
$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$



Binary tree

Example

$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$



In practise, branches are cut

Example

$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$

Branches and solutions

- Some branches can be cut : if $w_1 = w_2 = 0$ then $\alpha_1 = \alpha_1 + \alpha_3 = 0$ so $\alpha_1 = \alpha_3 = 0$ and α_2 free. No linear combination possible.
- Some leaves do not give a solution
- Some leaves are solutions

In practise, branches are cut

Example

$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$

Branches and solutions

- Some branches can be cut
- Some leaves do not give a solution : if $w_1 = w_3 = w_4 = 0$ and $w_2, w_5 \neq 0$, then $\alpha_1 = 0$ and $\alpha_2 = -\alpha_3 \neq 0$. w is not sparser than B_2 nor B_3: $2 = \mathcal{N}(w) = \mathcal{N}(B_2) = \mathcal{N}(B_3)$.
- Some leaves are solutions

In practise, branches are cut

Example

$$w = \sum_{i=1}^{3} \alpha_i B_i = (\alpha_1, \alpha_1 + \alpha_3, 2\alpha_1 + \alpha_2 + \alpha_3, \alpha_1, \alpha_1 + \alpha_2)$$

Branches and solutions

- Some branches can be cut
- Some leaves do not give a solution
- Some leaves are solutions : if $w_2 = w_3 = w_5 = 0$ and $w_1, w_4 \neq 0$, then $\alpha_3 = \alpha_2 = -\alpha_1 \neq 0$. Linear combination possible on row 1 as $\alpha_1 \neq 0$ and $2 = \mathcal{N}(w) < \mathcal{N}(B_1) = 5$.

End of example

Solution found

We have found a solution with $j = 1$ and $(\alpha_1, \alpha_2, \alpha_3) = (1, -1, -1)$, this corresponds to: $B_1 \leftarrow B_1 - B_2 - B_3$.

End of example

Solution found

We have found a solution with $j = 1$ and $(\alpha_1, \alpha_2, \alpha_3) = (1, -1, -1)$, this corresponds to: $B_1 \leftarrow B_1 - B_2 - B_3$.

Example

After this linear combination: $B = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$.

End of example

Solution found

We have found a solution with $j = 1$ and $(\alpha_1, \alpha_2, \alpha_3) = (1, -1, -1)$, this corresponds to: $B_1 \leftarrow B_1 - B_2 - B_3$.

Example

After this linear combination: $B = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$.

Second step: trying to find another linear combination

On the new matrix B, no row can be made sparser. Then, B is a sparsest basis.

Complexity for one row improvement

Notation

Consider a basis B of dimensions $m \times n$. Denote $d = \max\{\mathcal{N}(B_i), i \in [\![1, m]\!]\}$.

Complexity for one row improvement

Notation

Consider a basis B of dimensions $m \times n$. Denote $d = \max\{\mathcal{N}(B_i), i \in [\![1, m]\!]\}$.

Number of processed nodes (for one row improvement)

At most $\sum_{i=0}^{d} \binom{k}{i}$ nodes at depth k are reached when going to the right at most d times. One shows that the total number of processed nodes in the tree is bounded by: $\displaystyle\sum_{i=0}^{d} \binom{n+1}{i+1} \leq 2(n+1)^{d+1}$.

Complexity for one row improvement

Notation

Consider a basis B of dimensions $m \times n$. Denote $d = \max\{\mathcal{N}(B_i), i \in [\![1, m]\!]\}$.

Number of processed nodes (for one row improvement)

At most $\sum_{i=0}^{d} \binom{k}{i}$ nodes at depth k are reached when going to the right at most d times. One shows that the total number of processed nodes in the tree is bounded by: $\displaystyle\sum_{i=0}^{d} \binom{n+1}{i+1} \leq 2(n+1)^{d+1}$.

Remarks

We observed that models with small values of d were easily solved. The total number of row improvements is bounded by nd.

Table of Contents

Experiments on the 'BioModels' database

We study the curated models on the biomodels database: www.ebi.ac.uk/biomodels-main/.

bases of models with one compartment		
214		
already sparsest	can be improved	aborted after 2 days running
141	71	2

Most of systems of reactions have less than 50 conservation laws and less than 200 species.

Maple times on CSB algorithm

Model	Size of B	d	Time
068	3×8	4	0.06s
064	4×21	18	1.99s
183	4×67	61	36.50s
086	5×17	12	3.12s
336	5×18	7	0.30s
237	6×26	17	0.91s
431	6×27	15	10.93s
475	7×23	14	10.59s

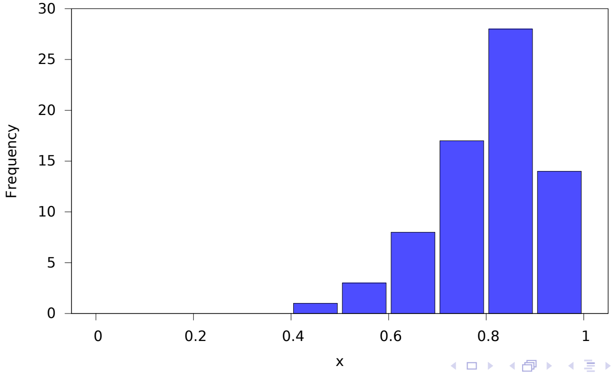
Model	Size of B	d	Time
014	8×86	45	235.9s
478	11×33	11	1.85s
153	11×75	38	964.6s
152	11×64	32	97.46s
334	13×73	50	132.6s
019	15×61	13	24.36s
332	25×166	49	$\approx$4000s
175	36×194	42	$\approx$1day

d: number of nonzeros in the row of B with the most of nonzeros
$(d = \max\{\mathcal{N}(B_i), i \in [\![1, m]\!]\})$

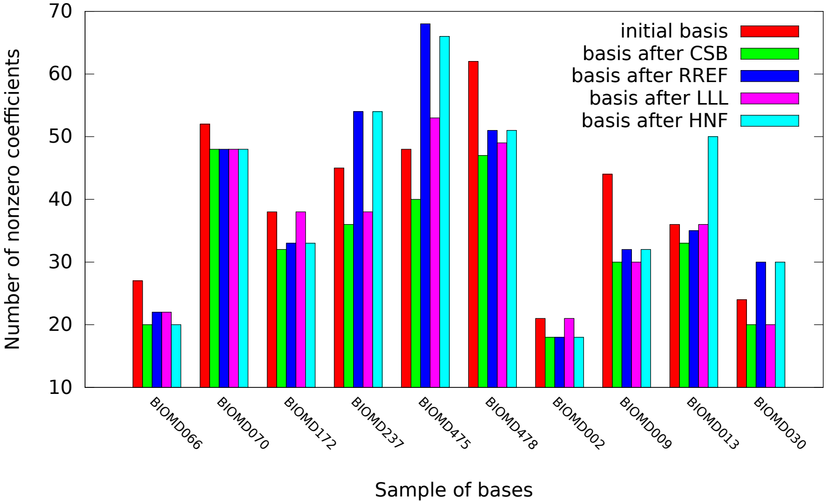
Improvement ratio on the 71 non sparsest bases

Improvement ratio

Consider an initial basis B and a sparsest basis B' computed by CSB for every model. We define the ratio $x = \frac{\mathcal{N}(B')}{\mathcal{N}(B)}$. For the 71 non sparsest bases, this ratio satisfies $0 < x < 1$.



Comparison of sparsity on exact algorithms



Questions/Suggestions

Any questions or suggestions ?