The FreeMABSys Project and the MABSys Library

François Lemaire, Aslı Ürgüplü.

University Lille 1 (Symbolic Computation Team)

Magix@Lix
École Polytechnique, Septembre 2011
Work supported by the ANR LEDA

Plan

Plan

Implementation

- MABSys: Modeling and Analysis of Biological Systems
- Authors : François Lemaire and Aslı Ürgüplü
- Coded in Maple (2008-)
- Download : `www.lifl.fr/~lemaire/MABSys`

Description

handles : models described by chemical reactions

provides : routines for reducing/simplifying the model (ODEs)

goal :

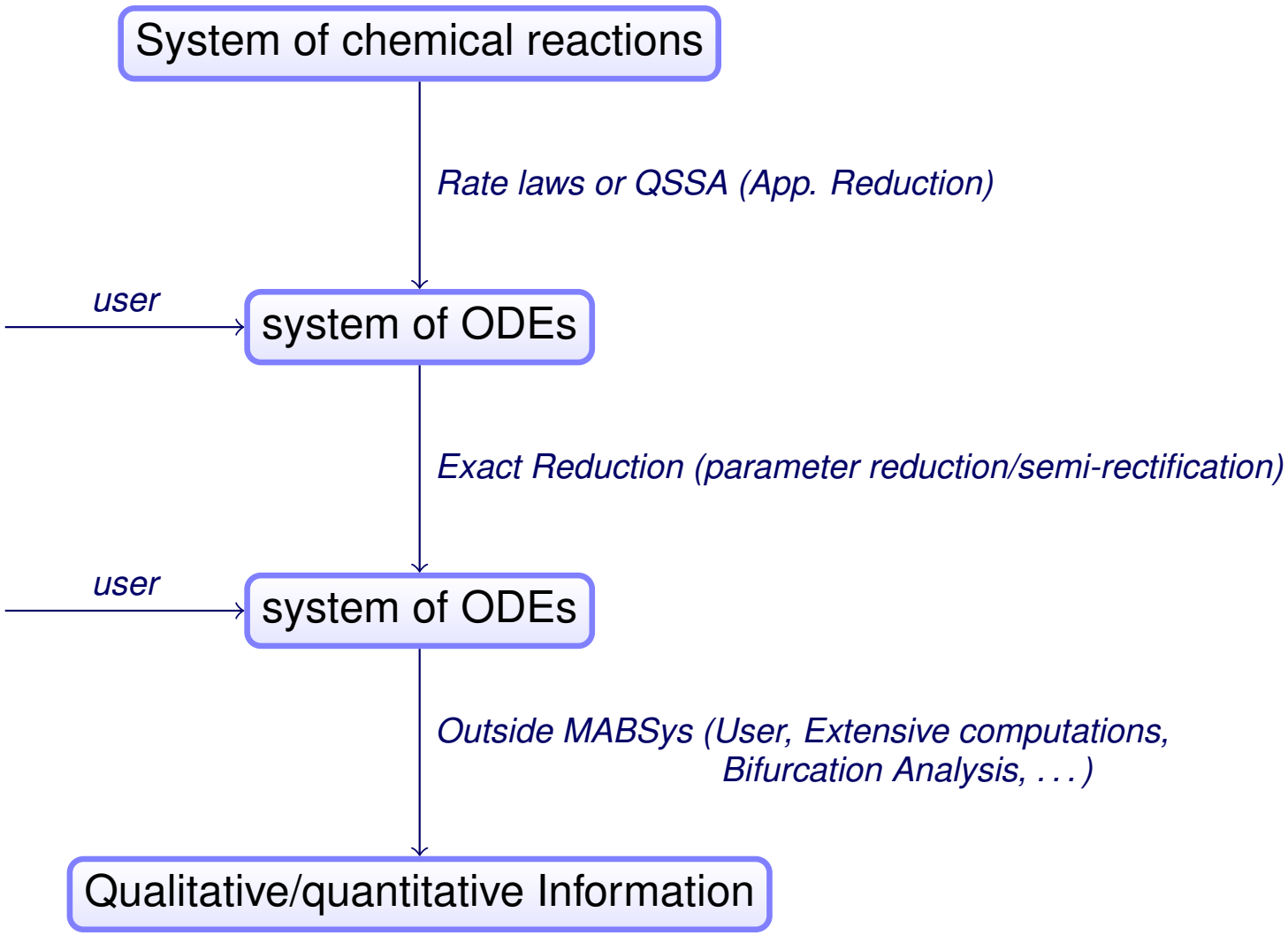
- make the reductions usually made by hand automatically
- help the analysis of the model

spirit : accessible to non specialists, inspired by collaboration with modelers

evolution :

- version of MABSys (in C) based on BLAD, and free software
- supported by the ANR LEDA

Typical use of MABSys



Techniques

- differential elimination for the QSSA

F. Boulier, M. Lefranc, F. Lemaire, and P.-E. Morant.

Model Reduction of Chemical Reaction Systems using Elimination.
MACIS, 2007.
http://hal.archives-ouvertes.fr/hal-00184558/fr.

- Lie Symmetries (parameter reduction, semi-rectification)

F. Lemaire and A. Ürgüplü.

A Method for Semi-Rectifying Algebraic and Differential Systems using Scaling type Lie Point Symmetries with Linear Algebra.
In *Proceedings of ISSAC*, 2010.

Olver, P. J.

Applications of Lie groups to differential equations, second ed., vol. 107 of *Graduate Texts in Mathematics*. Springer Verlag, 1993.

A. Sedoglavic.

Reduction of Algebraic Parametric Systems by Rectification of their Affine Expanded Lie Symmetries.
In K. Horimoto H. Anai and T. Kutsia, editors, *Proceedings of Algebraic Biology 2007*, volume 4545 of *LNCS*, pages 277–291, 2007.

Techniques are hidden to the user ! We want a friendly interface.

Plan

Basic enzymatic degradation

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

substrate S, product P, complex C, enzyme E

```
> R1 := NewReaction(E+S,C,MassActionLaw(k1),fast=true):
> R2 := NewReaction(C,E+S,MassActionLaw(km1),fast=true):
> R3 := NewReaction(C,E+P,MassActionLaw(k2)):

> RS := [R1,R2,R3]:
```

Remark : one can use `CustomizedLaw` for arbitrary rates

From the reactions to a dynamical system

$$
\begin{array}{lrcl}
R_1: & E + S & \xrightarrow{k_1} & C \\
R_2: & C & \xrightarrow{k_{-1}} & E + S \\
R_3: & C & \xrightarrow{k_2} & E + P
\end{array}
$$

Dynamical system: $\dot{X} = MV$

Vector of concentrations: $X = \begin{pmatrix} E \\ S \\ C \\ P \end{pmatrix}$

Stoichiometric matrix: $M = \begin{pmatrix} -1 & 1 & 1 \\ -1 & 1 & 0 \\ 1 & -1 & -1 \\ 0 & 0 & 1 \end{pmatrix}$

Vector of rates (following mass action law): $V = \begin{pmatrix} k_1 E S \\ k_{-1} C \\ k_2 C \end{pmatrix}$

Basic operations

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

Stoechiometry matrix

```
> StoichiometricMatrix(RS, [E,S,C,P]);
       [-1     1     1]
       [               ]
       [-1     1     0]
       [               ]
       [ 1    -1    -1]
       [               ]
       [ 0     0     1]
```

Vector of rates

```
> RateVector(RS);
       [k1 E S]
       [       ]
       [km1 C ]
       [       ]
       [ k2 C ]
```

Basic operations

Conversion to an ODE system

```
> ReactionSystem2ODEs(RS, [E,S,C,P]);
        d
       [-- E(t) = -k1 E(t) S(t) + km1 C(t) + k2 C(t),
        dt
        d
        -- S(t) = -k1 E(t) S(t) + km1 C(t),
        dt
        d
        -- C(t) = k1 E(t) S(t) - km1 C(t) - k2 C(t),
        dt
        d
        -- P(t) = k2 C(t)]
        dt
```

Steady points equations

```
> Equilibria(RS);
                          [k1 E S - km1 C - k2 C, k2 C]
```

Other basic routines: access reactions information, ODE simulation and plottings,. . .

Plan

Quasi Steady State Approximation (QSSA)

Idea

- if some reactions are tagged fast, one can assume the fast reactions are (almost) at the equilibria
- then, one computes a special ODE system which is compatible with the fast reactions being at equilibria
- it involves the introduction of dummy variables and differential elimination
- it is approximate : limit case where the fast reactions are infinitely fast

The `ModelReduce` routine

- input : RS (list of chemical reactions), X (list of concentrations), options
- output : a list of reduced systems

Our algorithm ModelReduce

- completely algorithmic (based on differential elimination)
- makes algorithmic result from [1, 2, 3] and most probably [5, 6]


[1] V. Van Breusegem and G. Bastin.
Reduced order dynamical modelling of reaction systems: a singular perturbation approach
30th IEEE Conf. on Decision and Control. pp. 1049-1054, 1991.


[2] N. Vora and P. Daoutidis.
Nonlinear model reduction of chemical reaction systems.
AIChE Journal vol. 47, pp. 2320-2332, 2001.


[3] M. Bennet, D. Volfson, L. Tsimring and J. Hasty.
Transient Dynamics of Genetic Regulatory Networks
Biophysical Journal vol. 92, pp. 3501-3512, 2007


[4] F. Boulier, M. Lefranc, F. Lemaire and P.-E. Morant.
Model Reduction of Chemical Reaction Systems using Elimination.
MACIS, http://hal.archives-ouvertes.fr/hal-00184558, 2007.


[5] Nöthen, Anna Lena, PhD Thesis


[6] Schauer und Heinrich
Quasi-steady-state approximation in the mathematical modeling of biochemical reaction networks

Quasi-steady state approximation

The usual reduction

- $\dot{S}(t) = -\dfrac{V_m\, S(t)}{K + S(t)}$ assuming $S \gg E_0$
- $V_m = k_2\, E_0$
- Briggs-Haldane: $K = \frac{k_{-1}}{k_1}$
- Henri-Michaëlis-Menten: $K = \frac{k_{-1}+k_2}{k_1}$

By Hand

Our reduction (same as in section 5.2.3 in Anna Lena Nöthen PhD Thesis)

- $\dot{S} = -\dfrac{V_m\, S\,(K + S)}{K\, E_0 + (K + S)^2}$ seems valid even if $S < E_0$
- $V_m = k_2\, E_0$
- $K = \frac{k_{-1}}{k_1}$

Automatic

Quasi-steady state approximation

The usual reduction

- $\dot{S}(t) = -\dfrac{V_m\, S(t)}{K + S(t)}$ assuming $S \gg E_0$
- $V_m = k_2\, E_0$
- Briggs-Haldane: $K = \frac{k_{-1}}{k_1}$
- Henri-Michaëlis-Menten: $K = \frac{k_{-1}+k_2}{k_1}$

By Hand

Our reduction (same as in section 5.2.3 in Anna Lena Nöthen PhD Thesis)

- $\dot{S} = -\dfrac{V_m\, S\,(K + S)}{K\, E_0 + (K + S)^2}$ seems valid even if $S < E_0$
- $V_m = k_2\, E_0$
- $K = \frac{k_{-1}}{k_1}$

Automatic

Detail of the reduction

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

Step 1: build the system

$$\begin{aligned} \dot{C} &= F_1 - k_2 C \\ \dot{S} &= -F_1 \\ \dot{E} &= -F_1 + k_2 C \\ \dot{P} &= k_2 C \\ k_1 ES &= k_{-1} C \qquad (eq) \end{aligned}$$

- F_1: contribution of the fast reaction (unknown for the moment)
- (eq) implies that $E + S \rightleftharpoons C$ is at equilibria

Detail of the reduction

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

Step 2: eliminate the F_1 with Rosenfeld–Gröbner

$$\begin{aligned} \dot{S} &= -\frac{k_2 k_1 ES(k_1 S + k_{-1})}{k_{-1}(k_{-1} + k_1 S + k_1 E)} \\ \dot{C} &= \dots \\ \dot{S} &= \dots \\ \dot{E} &= \dots \\ F_1 &= \dots \end{aligned}$$

Detail of the reduction

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

Step 3: use conservations laws

$$\begin{aligned} \dot{S} &= -\frac{k_1 k_2 E_0 S (k_1 S + k_{-1})}{k_1 k_{-1} E_0 + (k_1 S + k_{-1})^2} \\ \dot{C} &= \dots \\ \dot{S} &= \dots \\ \dot{E} &= \dots \end{aligned}$$

using the conservation laws:

- $C + P + S = C_0 + P_0 + S_0$
- $C + E = C_0 + E_0$
- and assuming $C_0 = 0$.

Detail of the reduction

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \qquad \text{(supposed fast)}$$

$$C \xrightarrow{k_2} E + P \qquad \text{(supposed slow)}$$

Step 4: use symmetries

$$\dot{S} = -\frac{k_1 k_2 E_0 \, S \, (k_1 S + k_{-1})}{k_1 k_{-1} \, E_0 + (k_1 S + k_{-1})^2}$$

is reformulated to

$$\dot{S} = -\frac{V_m \, S \, (K + S)}{K \, E_0 + (K + S)^2}$$

with $V_m = k_2 \, E_0$ and $K = \frac{k_{-1}}{k_1}$

no conservation laws

```
> output := ModelReduce(RS, [E,C,P,S]): # only one output
> output[1][1];
                                2
 d                    km1 C(t)   k2
[-- E(t) = ----------------------------,
 dt                              2
           km1 S(t) + k1 S(t)   + km1 C(t)

                                    2
    d                      km1 C(t)   k2             d
    -- C(t) = - ----------------------------,  -- P(t) = k2 C(t),
    dt                              2             dt
                km1 S(t) + k1 S(t)   + km1 C(t)

    d              S(t) (km1 + k1 S(t)) k2 C(t)
    -- S(t) = - ----------------------------]
    dt                              2
                km1 S(t) + k1 S(t)   + km1 C(t)

# QSSA Assumptions:
> output[1][2];
                             [k1 E(t) S(t) - km1 C(t)]
```

with conservation laws

```
output := ModelReduce(RS, [E,C,P,S], useConservationLaws=true):
red_sys := output[1][1]:
red_sys := subs(C_0=0, red_sys):
red_sys[4];
```

$$\frac{d}{dt} S(t) = -\frac{E_0\, k2\, k1\, S(t)\, (km1 + k1\, S(t))}{k1\, S(t)^2 + 2\, S(t)\, km1\, k1 + km1\, k1\, E_0 + km1^2}$$

- one has a diff. equation in $S(t)$ only, but still many parameters.
- we are far from:

$$\dot{S}(t) = -\frac{V_m\, S(t)}{K + S(t)}$$

with $V_m = k_2\, E_0$ and $K = \frac{k_{-1}}{k_1}$ (or $\frac{k_{-1}+k_2}{k_1}$)

Plan

Exact Reductions

Idea

- obtain an **equivalent** system which is "easier" to study, by operating a change of variables

Two routines

- `InvariantizeByScalings` : reduce the number of parameters
  - reduce the complexity of parameter value exploration
  - helps hand analysis
  - (shown on the enzymatic degradation)
- `SemiRectifySteadyPoints` : reduce the number of parameters on which the steady points depend
  - some parameters only affect the dynamic

Techniques

- one restricts to monomial maps ($x_i \to x_1^{\alpha_1} \cdots x_n^{\alpha_n}$). Ex: $V_m \to V_m/k_1$
  $\to$ ensures equivalence of systems, and positivity of parameters
- based on ELPS (Maple package by A. Sedoglavic) for symmetry computations

InvariantizeByScalings

- Input: a dynamical system, list of positive variables, list of remaining variables.
- Output: a reduced system, a change of variables, list of parameters removed

```
> red_sys[4];
          d                   E_0 k2 k1 S(t) (k1 S(t) + km1)
          -- S(t) = - ----------------------------------------------
          dt              2      2                                 2
                      k1   S(t)   + 2 S(t) km1 k1 + km1 k1 E_0 + km1

> output := InvariantizeByScalings(red_sys, [k1,km1,k2], [],
>                                  fixedvars=[E,C,P,S],
>                                  scaletime=false):
> red_sys2 := output[1]:
> red_sys2[4];
               d                S(t) k2 E_0 (S(t) + km1)
               -- S(t) = - -----------------------------------
               dt                2                             2
                           S(t)   + 2 km1 S(t) + E_0 km1 + km1

> output[2], output[3];
                                    km1
                              [km1 = ---], [k1]
                                    k1
```

A last hypothesis

- If one assumes $S >> E_0$, one retrieves a simpler formula (classical hypothesis)

```
> red_sys2[4];
                d                    S(t) k2 E_0 (S(t) + km1)
                -- S(t) = - ------------------------------------
                dt                              2                   2
                             2 km1 S(t) + S(t)   + E_0 km1 + km1

> ApproximateSmallQuantity(red_sys2[4], E_0<S(t));
                            d            S(t) k2 E_0
                            -- S(t) = - -----------
                            dt           S(t) + km1
```

Better simplification ?

- Another step can automatically rename `k2 E_0` into `k2`.
- We simplify by reducing the **number** of parameters.
- Further work : find changes of variables which reduce the **size** of the equations ($\dot{x}(t) = 1 + a + a\,b\,x(t)$ into $\dot{x}(t) = 1 + a + b\,x(t)$)

A few words on symmetries

- the equation

$$\dot{S}(t) = \frac{S(t)\, k_1\, k_2\, (k_1\, S(t) + k_{-1})}{k_1^2\, S(t)^2 + 2\, k_1\, k_{-1}\, S(t) + k_1\, k_{-1}\, E_0 + k_{-1}^2}$$

  is left invariant by the transformation:

$$\begin{array}{rcl} k_1 & \rightarrow & \lambda k_1 \\ k_{-1} & \rightarrow & \lambda k_{-1} \end{array}$$

- this suggests the equation depends on the ratio k_1/k_{-1}.
- by performing $k_{-1} \rightarrow k_{-1} k_1$, the equation becomes:

$$\dot{S}(t) = \frac{S(t)\, k_2\, (S(t) + k_{-1})}{S(t)^2 + 2\, k_{-1}\, S(t) + k_{-1}\, E_0 + k_{-1}^2}$$

SemiRectifySteadyPoints

- Input: a dynamical system, list of positive variables to remove from the steady points, list of remaining variables
- Output: a reduced system, its steadypoints, a change of variables, the parameters made free, list of parameters removed

```
> red_sys;
        d                                  2        d                          2
       [-- x(t) = 1 - x(t) + k2 x(t)  y(t), -- y(t) = b - k2 x(t)  y(t)]
        dt                                 dt

> # the steady points depends on b and k2
> SteadyPointSystem(red_sys);
                                    2            2
                        [1 - x + k2 x  y, b - k2 x  y]

> output := SemiRectifySteadyPoints(red_sys, [b,k2], [x,y]):
> # the steady points depends on b only
> output[1][2];
                                   2         2
                          [1 - x + x  y, b - x  y]
```

SemiRectifySteadyPoints (II)

```
> # reparametrized system
> output[1][1];
        d                              2        d                              2
       [-- x(t) = 1 - x(t) + x(t)  y(t), -- y(t) = k2   (b - x(t)  y(t))   ]
        dt                               dt

> # the steady points depends on b only
> output[1][2];
                                    2          2
                          [1 - x + x  y, b - x  y]
> # change of variable
> output[1][3];
                                  [y = y k2]

> # freed parameter
> output[1][4];
                                     [k2]
```

- only b affects the steady-points
- only k_2 affects the dynamics around the steady-point

A few words on semi-rectification

- the differential system

$$\begin{aligned} \dot{x}(t) &= 1 - x(t) + k_2\, x(t)^2\, y(t) \\ \dot{y}(t) &= b - k_2 x(t)^2 y(t) \end{aligned}$$

  has no (scaling) symmetry, but its steady point

$$\begin{aligned} 0 &= 1 - x + k_2\, x^2\, y \\ 0 &= b - k_2\, x^2\, y \end{aligned}$$

  has one:

$$k_2 \to \lambda\, k_2, \quad y \to y/\lambda$$

- trick: use the symmetry of the steady point on the differential system, using the change of variable $y \to y/k_2$
-

$$\begin{aligned} \dot{x}(t) &= 1 - x(t) + x(t)^2\, y(t) \\ \dot{y}(t) &= k_2(b - x(t)^2 y(t)) \end{aligned}$$

Plan

The example

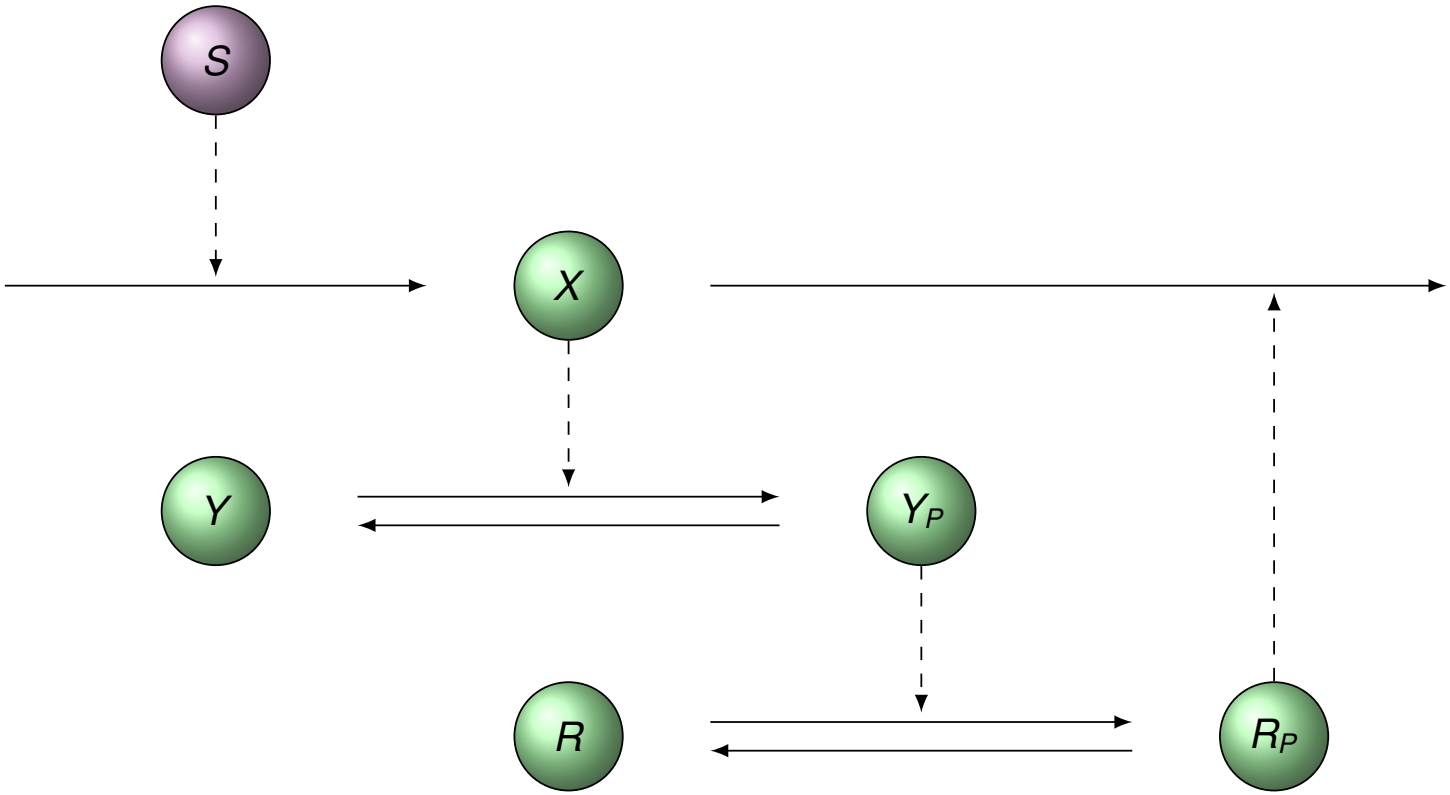


Figure: Tyson's negative feedback oscillator.

- S is the signal, X, Y, R, Y_P and R_P are other components of the signaling network.
- The negative feedback loop is closed by R_P activating the degradation of X.

The Associated ODE system

$$\begin{cases} \dfrac{\mathrm{d}X}{\mathrm{d}t} = k_0 + k_1\, S - k_2\, X - k_2'\, R_P\, X, \\ \dfrac{\mathrm{d}Y_P}{\mathrm{d}t} = \dfrac{k_3\, X\,(Y_T - Y_P)}{K_{m3} + Y_T - Y_P} - \dfrac{k_4\, Y_P}{K_{m4} + Y_P}, \\ \dfrac{\mathrm{d}R_P}{\mathrm{d}t} = \dfrac{k_5\, Y_P\,(R_T - R_P)}{K_{m5} + R_T - R_P} - \dfrac{k_6\, R_P}{K_{m6} + R_P} \end{cases}$$

where $Y_T = Y + Y_P$ and $R_T = R + R_P$ are total concentrations (constants).

Number of variables

- 15 parameters
- 15 parameters affect the steady points
- 3 concentrations

Exact simplification of the model (I)

Removing parameters (`InvariantizeByScalings`)

```
> out := InvariantizeByScalings(ODEs,[k1,k3,k5,Km3,Km5,k4,k6,k2p,\
                                      Km4,Km6,k2,YT,RT,S],[X,YP,RP,k0]):
memory used=206.0MB, alloc=12.2MB, time=4.18
.........................
memory used=259.5MB, alloc=12.3MB, time=5.05
> CoC1 := out[2]:
> FreedParams1 := out[3];
                    FreedParams1 := [k1, k3, k5, Km3, Km5]
```

Number of variables

- 10 parameters = 15 (original) - 5 (freed)
- 10 parameters affect the steady points
- 3 concentrations

Exact simplification of the model (II)

SemiRectification (`SemiRectifySteadyPoints`)

```
out := SemiRectifySteadyPoints(InterODEs,[k2p,k2,Km4,Km6,k4,k6,YT,RT,S],\
                                         [X,YP,RP,k0]):
FinalODEs := out[1,1]:
CoC2 := out[1,3]:
FreedParams2 := out[1,4];
                              FreedParams2 := [k2p, k4]
```

Number of variables

- 10 parameters
- 8 parameters (10 - 2) affect the steady points
- 3 concentrations

Final version

$$
\begin{cases}
\dfrac{\mathrm{d}\widetilde{X}}{\mathrm{d}\widetilde{t}} = \left(\widetilde{k}_0 + \widetilde{S} - \widetilde{k}_2\,\widetilde{X} - \widetilde{R}_P\,\widetilde{X}\right)\widetilde{k}_2', \\
\dfrac{\mathrm{d}\widetilde{Y}_P}{\mathrm{d}\widetilde{t}} = \left(\dfrac{\widetilde{X}\left(\widetilde{Y}_T - \widetilde{Y}_P\right)}{1 + \widetilde{Y}_T - \widetilde{Y}_P} - \dfrac{\widetilde{Y}_P}{\widetilde{K}_{m4} + \widetilde{Y}_P}\right)\widetilde{k}_4, \\
\dfrac{\mathrm{d}\widetilde{R}_P}{\mathrm{d}\widetilde{t}} = \dfrac{\widetilde{Y}_P\left(\widetilde{R}_T - \widetilde{R}_P\right)}{1 + \widetilde{R}_T - \widetilde{R}_P} - \dfrac{\widetilde{k}_6\,\widetilde{R}_P}{\widetilde{K}_{m6} + \widetilde{R}_P}.
\end{cases}
$$

Change of coordinate

$$
\begin{array}{lllll}
\widetilde{k}_4 = \frac{k_4\,K_{m5}}{K_{m3}^2\,k_5}, & \widetilde{k}_6 = \frac{k_6}{K_{m3}\,k_5}, & \widetilde{k}_2' = \frac{k_2'\,K_{m5}^2}{K_{m3}\,k_5}, & \widetilde{K}_{m4} = \frac{K_{m4}}{K_{m3}}, & \widetilde{K}_{m6} = \frac{K_{m6}}{K_{m5}}, \\
\widetilde{k}_0 = \frac{k_0\,k_3}{K_{m5}\,k_2'\,k_4}, & \widetilde{k}_2 = \frac{k_2}{K_{m5}\,k_2'}, & \widetilde{Y}_T = \frac{Y_T}{K_{m3}}, & \widetilde{R}_T = \frac{R_T}{K_{m5}}, & \widetilde{S} = \frac{S\,k_1\,k_3}{K_{m5}\,k_2'\,k_4}, \\
\widetilde{t} = \frac{t\,K_{m3}\,k_5}{K_{m5}}, & \widetilde{X} = \frac{X\,k_3}{k_4}, & \widetilde{Y}_P = \frac{Y_P}{K_{m3}}, & \widetilde{R}_P = \frac{R_P}{K_{m5}}, & \\
\widetilde{k}_1 = k_1, & \widetilde{k}_3 = k_3, & \widetilde{K}_{m3} = K_{m3}, & \widetilde{k}_5 = k_5, & \widetilde{K}_{m5} = K_{m5}.
\end{array}
$$

Plan

Conclusion

Further development

- free version, probably in C, based on BLAD (F. Boulier)
- a prototype in C implements QSSA and SBML connectivity
- supported by the ANR (french research agency) LEDA
  - LEDA : Logistique des Équations Différentielles Algébriques.
  - DAE naturally arise with the QSSA
  - main actors : Jean-Claude Yakoubsohn, Jean-Pierre Belaud, François Ollivier, François Boulier, JvdH. . .
- new techniques to develop
  - reparametrization to reduce the size of the systems
  - inspiration by forthcoming collaboration with modelers
  - . . .

Implementation Issue

Libraries needed

- differential elimination : BLAD (C), Differential Algebra (Maple)
- linear algebra on rational : not an issue
- linear algebra on polynomials : basic operations in BLAD
- SBML (XML specification for systems in biology) : libSBML (C, . . .)
- symmetry computations : ELPS (Maple), based on linear algebra and differentiation of polynomials
- many simple Maple operations can become cumbersome in C : use of rational exponents with BLAD ?

Interface

- C : can be integrated in other software
- Maple, MatheMagix : easy to use interactively