Algèbre Différentielle et Modélisation en Biologie Cours d'ouverture à l'École Jeunes Chercheurs « Modélisation Formelle des Réseaux de Régulation de Gènes »

François Boulier

4 juin 2010

1 Introduction

L'algèbre différentielle est une théorie algébrique des équations différentielles, comme l'indique le titre du premier livre de Ritt sur le sujet [18]. Les grands livres de la théorie sont [19] et [13]. L'élimination différentielle est une branche de l'algèbre différentielle. Les bases en ont été posées par Ritt et Seidenberg [22]. Sans oublier une contribution essentielle de Rosenfeld [20] pour les systèmes aux dérivées partielles, mais qui ne seront pas abordés dans ce cours.

L'élimination différentielle peut être vue comme une théorie rigoureuse de la simplification des systèmes d'équations différentielles polynomiales (donc non linéaires) dépendant éventuellement de paramètres. Dans ce cours, on la présente à travers le nouveau paquetage *DifferentialAlgebra* du logiciel MAPLE 14, qui a été développé par Edgardo Cheb-Terrab (MAPLESOFT) et par l'auteur.

Pour utiliser *DifferentialAlgebra*, il faut bien sûr disposer de MAPLE 14, qui est un logiciel commercial. Toutefois, tous les calculs effectués par le paquetage sont en fait réalisés par les bibliothèques BLAD [3], qui sont des bibliothèques *open source*, portables, écrites en langage C. Un exemple en BLAD, sans MAPLE, est fourni en section 6. L'intégration de BLAD dans des logiciels libres comme MATHEMAGIX ou SAGE est à l'étude. L'auteur serait très intéressé par l'intégration de BLAD dans des logiciels d'aide à la modélisation en biologie.

2 Réduction de modèles et élimination différentielle

L'élimination différentielle permet d'effectuer de la *réduction de modèles*. Le point de départ est un système de réactions chimiques généralisées [9]. Ce type de système est couramment utilisé pour modéliser des phénomènes beaucoup plus généraux que les susbtances chimiques réagissant dans une éprouvette [17]. On peut s'en servir, en particulier, pour modéliser des

réseaux de régulation de gènes. On leur associe un modèle déterministe, en appliquant la *loi d'action de masse*. On obtient ainsi un système d'équations différentielles polynomiales dépendant de paramètres. Les variables du systèmes sont des fonctions du temps, représentant les concentrations des différentes espèces chimiques. Il y a une équation différentielle par variable, et donc par espèce chimique. Les constantes de vitesse des réactions apparaissent dans les équations, en tant que paramètres. En général, ces *modèles déterministes naturels* sont trop gros pour être utiles. La réduction de modèles a pour but d'en construire des approximations, valables sous certaines hypothèses, mais plus simples à analyser.

Dans ce cours, on illustre une méthode de réduction, fondée sur la théorie de l'approximation d'état quasi-stationnaire [12]. À titre d'exemple, on étudie la réduction d'un modèle de réaction enzymatique, due à Henri, Michaelis et Menten au début du XXème siècle [8, 16]. Le système considéré est

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} ES \xrightarrow{k_2} E + P.$$

2.1 Le modèle déterministe naturel

Voici le modèle déterministe naturel, construit avec le paquetage *LinearAlgebra* de MAPLE. On trouve quatre espèces chimiques : l'enzyme E, le substrat S, le complexe intermédiaire *ES* et le produit P, auxquelles on associe quatre fonctions du temps, représentant leur concentration ; et trois constantes cinétique k_1, k_{-1} et k_2. On note X le vecteur des concentrations, N la matrice des coefficients stœchiométriques (une ligne par espèce, une colonne par réaction) et V le vecteur des vitesses.

```
with (LinearAlgebra):
X := <E(t), S(t), ES(t), P(t)>:
V := <k[1]*E(t)*S(t), k[-1]*ES(t), k[2]*ES(t)>:
N := <<-1, -1, 1, 0> | <1, 1, -1, 0> | <1, 0, -1, 1>>:
X, N, V;
```

$$\begin{bmatrix} E(t) \\ S(t) \\ ES(t) \\ P(t) \end{bmatrix}, \begin{bmatrix} -1 & 1 & 1 \\ -1 & 1 & 0 \\ 1 & -1 & -1 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} k_1\, E(t)\, S(t) \\ k_{-1}\, ES(t) \\ k_2\, ES(t) \end{bmatrix}$$

```
natsys0 := map (diff, X, t) = N . V;
```

$$natsys0 := \begin{bmatrix} \frac{d}{dt} E(t) \\ \frac{d}{dt} S(t) \\ \frac{d}{dt} ES(t) \\ \frac{d}{dt} P(t) \end{bmatrix} = \begin{bmatrix} -k_1\, E(t)\, S(t) + k_{-1}\, ES(t) + k_2\, ES(t) \\ -k_1\, E(t)\, S(t) + k_{-1}\, ES(t) \\ k_1\, E(t)\, S(t) - k_{-1}\, ES(t) - k_2\, ES(t) \\ k_2\, ES(t) \end{bmatrix}$$

2.2 Premiers pas en algèbre différentielle

La présentation qui suit est informelle. Le lecteur intéressé peut (mais ce n'est pas nécessaire), jeter un œil en section 7.

Du point de vue de l'algèbre différentielle, une expression comme

$$\frac{\mathrm{d}}{\mathrm{d}t}E(t) + k_1\,E(t)\,S(t) - k_{-1}\,ES(t) - k_2 ES(t)$$

est un polynôme différentiel, c'est-à-dire un polynôme, au sens habituel du terme, construit sur l'alphabet infini des *dérivées* d'ordre quelconque des variables dépendantes $E(t)$, $S(t)$, $ES(t)$ et $P(t)$. Les paramètres k_1, k_{-1} et k_2 peuvent être vus comme des éléments du corps des coefficients. Mais on peut aussi les considérer comme trois variables dépendantes supplémentaires, contraintes par les relations

$$\frac{\mathrm{d}}{\mathrm{d}t}k_1 = \frac{\mathrm{d}}{\mathrm{d}t}k_{-1} = \frac{\mathrm{d}}{\mathrm{d}t}k_2 = 0.$$

Le second point de vue est plutôt plus général que le premier. Le paquetage *DifferentialAlgebra* autorise les deux. On verra comment les utiliser pour contrôler les discussions de cas, effectuées par le simplificateur de systèmes de polynômes différentiels, *RosenfeldGroebner*.

La notion de *classement* (*ranking* en Anglais) est essentielle dans toute la suite du cours. En effet, les classements fournissent des critères que le simplificateur *RosenfeldGroebner* cherche à optimiser.

Formellement, un classement est un relation d'ordre total sur l'ensemble infini des dérivées. Supposons qu'un classement soit fixé. Tout polynôme différentiel admet alors une *dérivée dominante* (parmi toutes les dérivées figurant dans l'écriture du polynôme, c'est la plus grande, au sens du classement). Pour le simplificateur de systèmes de polynômes différentiels, un polynôme différentiel A est considéré comme plus simple qu'un autre, B, si la dérivée dominante de A est plus petite que celle de B, au sens du classement.

Dans *DifferentialAlgebra*, les classements se définissent grâce à la fonction *DifferentialRing*, en spécifiant une liste de *blocs*. Les plus simples des classements sont les classements d'élimination. La liste de blocs est une simple liste de variables dépendantes. Sur l'exemple, $E(t)$ et toutes ses dérivées sont plus grandes que $S(t)$ et toutes ses dérivées, qui sont elles-mêmes plus grandes que $ES(t)$ et toutes ses dérivées, qui sont elles-mêmes plus grandes que $P(t)$ et toutes ses dérivées.

```
with (DifferentialAlgebra):
R := DifferentialRing (derivations = [t], blocks = [E,S,ES,P]);
```

$$R := \mathit{differential\_ring}$$

Une liste de dérivées.

```
L := [E(t),S(t),ES(t),P(t),diff(E(t),t),diff(S(t),t),diff(P(t),t,t)];
```

$$L := \left[E(t),\ S(t),\ ES(t),\ P(t),\ \frac{\mathrm{d}}{\mathrm{d}t}E(t),\ \frac{\mathrm{d}}{\mathrm{d}t}S(t),\ \frac{\mathrm{d}^2}{\mathrm{d}t^2}P(t)\right]$$

La liste, triée par ordre décroissant, au sens du classement d'élimination défini dans R.

```
Tools:-SortByRank (L, descending, R);
```

$$\left[\frac{d}{dt}E(t), E(t), \frac{d}{dt}S(t), S(t), ES(t), \frac{d^2}{dt^2}P(t), P(t)\right]$$

Les classements compatibles avec l'ordre total (*orderly* en Anglais) se définissent en utilisant un deuxième niveau de crochets. La liste de blocs contient un unique bloc. Plus une dérivée est d'ordre élevé, plus elle est grande. Deux dérivées de même ordre sont comparées en fonction de leur position dans le bloc. Sur l'exemple, la dérivée d'ordre deux est la plus grande, suivent les deux dérivées d'ordre un, puis les quatre dérivées d'ordre zéro.

```
R := DifferentialRing (derivations = [t], blocks = [[E,S,ES,P]]):
Tools:-SortByRank (L, descending, R);
```

$$\left[\frac{d^2}{dt^2}P(t), \frac{d}{dt}E(t), \frac{d}{dt}S(t), E(t), S(t), ES(t), P(t)\right]$$

On a souvent besoin de combiner les deux types de classement. On a affaire à un classement d'élimination par blocs. Sur l'exemple, $E(t)$, $S(t)$ et toutes leurs dérivées sont plus grandes que $ES(t)$, $P(t)$ et toutes leurs dérivées. Deux dérivées de variables dépendantes listées dans un même bloc, sont comparées suivant la méthode précédente.

```
R := DifferentialRing (derivations = [t], blocks = [[E,S],[ES,P]]):
Tools:-SortByRank (L, descending, R);
```

$$\left[\frac{d}{dt}E(t), \frac{d}{dt}S(t), E(t), S(t), \frac{d^2}{dt^2}P(t), ES(t), P(t)\right]$$

2.3 Le modèle déterministe naturel, avec *DifferentialAlgebra*

La principale fonctionnalité fournie par le paquetage *DifferentialAlgebra* est un simplificateur, nommé *RosenfeldGroebner*, dont la théorie est décrite dans [4, 5]. Ce simplificateur prend en entrée deux paramètres : un système d'équations différentielles $\mathscr{S}$ à simplifier ainsi qu'un classement. Il retourne une *chaîne différentielle régulière* (on dit aussi un *ensemble caractéristique*), c'est-à-dire un système d'équations différentielles simplifié, équivalent à $\mathscr{S}$. Comme les systèmes manipulés sont non linéaires, l'algorithme peut être amené à effectuer une discussion de cas, et à retourner une liste de chaînes différentielles régulières, au lieu d'une seule.

Les chaînes différentielles régulières sont définies, et calculées, en fonction du classement. En particulier, le simplificateur s'est efforcé de calculer les équations dont les dérivées dominantes sont les plus petites possibles, au sens du classement.

Dans le classement ci-dessous, les trois constantes de vitesse sont regroupées dans le bloc le plus petit. Les parenthèses vides qui les suivent indiquent qu'on a affaire à des constantes, au sens mathématique, c'est–à–dire des quantités dont la dérivée est nulle. Les quatre variables représentant les concentrations sont groupées dans un même bloc. On indique ainsi au logiciel qu'on cherche des équations qui contiennent des dérivées de ces quatre concentrations, d'ordre aussi peu élevé que possible.

```
with (DifferentialAlgebra);
```

[*BelongsTo, ChangeRanking, DifferentialRing, Equations, Get, Inequations, Is, NormalForm, PowerSeriesSolution, ReducedForm, RosenfeldGroebner, Tools*]

Le résultat de l'appel de fonction est une table qui encapsule les données.

```
R := DifferentialRing
         (derivations = [t],
          blocks = [[E,S,ES,P], [k[1](),k[-1](),k[2]()]]);
```

$$R := differential\_ring$$

Le modèle déterministe naturel ainsi que le classement encapsulé dans R sont fournis en paramètres au simplificateur. La liste retournée ne contient qu'une seule chaîne différentielle régulière.

```
natsys := [ seq (lhs (natsys0) [i] = rhs (natsys0) [i],
                 i = 1 .. Dimension (X)) ]:
natideal := RosenfeldGroebner (natsys, R);
```

$$natideal := [regular\_differential\_chain]$$

La chaîne différentielle régulière est elle-même encapsulée dans une table. La fonction *Equations* permet d'en consulter le contenu. Avec le mot-clef *solved*, la *dérivée dominante* de chaque équation apparaît en partie gauche de l'équation.

```
Equations (natideal[1], solved);
```

$$\left[\frac{d}{dt}E(t) = -k_1\,E(t)\,S(t) + k_{-1}\,ES(t) + k_2\,ES(t),\ \frac{d}{dt}S(t) = -k_1\,E(t)\,S(t) + k_{-1}\,ES(t),\ \frac{d}{dt}ES(t) = k_1\,E(t)\,S(t) - k_{-1}\,ES(t) - k_2\,ES(t),\ \frac{d}{dt}P(t) = k_2\,ES(t)\right]$$

2.4 Première digression

Les équations du modèle déterministe naturel n'ont pas été modifiées par le simplificateur. En effet, tout système dynamique est une chaîne différentielle régulière, vis-à-vis de tout classement compatible avec l'ordre total, sur les variables dépendantes. Toutefois, si on change le classement, la chaîne change. Le classement défini dans la variable R_2 est différent de celui de défini dans R. En effet, les quatre concentrations ne sont plus groupées dans un même bloc. On indique ainsi au logiciel qu'on cherche des équations qui ne dépendent que de $P(t)$ et de ses dérivées. Peu importe leur ordre de dérivation.

```
R2 := DifferentialRing
           (derivations = [t],
            blocks = [E, S, ES, P, [k[1](),k[-1](),k[2]()]]);
```

$$R2 := differential\_ring$$

La fonction *ChangeRanking* applique une variante de *RosenfeldGroebner*, spécialisée pour les changements de classement [7, Algorithme PARDI].

```
natideal2 := ChangeRanking (natideal[1], R2);
```

$$natideal2 := regular\_differential\_chain$$

Les équations de la chaîne différentielle régulière $natideal_2$ sont trop volumineuses pour être imprimées. Les points de suspension cachent une vingtaine de lignes.

```
Equations (natideal2, solved);
```

$$\left[E(t) = -\left(-\left(\frac{d^4}{dt^4} P(t) \right) \left(\frac{d^2}{dt^2} P(t) \right) k_2 + \left(\frac{d^3}{dt^3} P(t) \right)^2 k_2 + 2 \left(\frac{d^2}{dt^2} P(t) \right)^3 k_1 \right.$$

$$\vdots$$

$$\left. + \left(\frac{d^4}{dt^4} P(t) \right) \left(\frac{d^3}{dt^3} P(t) \right) \left(\frac{d^2}{dt^2} P(t) \right) \left(\frac{d}{dt} P(t) \right) k_2^4 + 4 \left(\frac{d^2}{dt^2} P(t) \right)^6 k_1^2 + \left(\frac{d^3}{dt^3} P(t) \right)^4 k_2^2 \right) \right]$$

La fonction *LeadingRank* du sous-paquetage *Tools*, permet de ne visualiser que les dérivées dominantes (élévées à leur degré le plus élevé). On voit que la dérivée dominante de l'une des équations est d'ordre 4 en $P(t)$ (et de degré 2 en la dérivée quatrième de $P(t)$). Au vu du classement utilisé, cette équation ne dépend que de $P(t)$ et de ses dérivées d'ordre 0 à 3. Elle ne dépend pas du tout des trois autres concentrations.

```
Tools:-LeadingRank (natideal2);
```

$$\left[\left(\frac{d^4}{dt^4} P(t) \right)^2, ES(t), S(t), E(t) \right]$$

3 Une approximation trop violente

Le texte qui suit est extrait de la page anglaise de la Wikipedia, expliquant la réduction de Michaelis-Menten.

> *The validity of the following derivation rests on the reaction scheme given below and two key assumptions : that the total enzyme concentration and the concentration of the intermediate complex do not change over time.*
>
> [...]
>
> *The first key assumption in this derivation is the quasi-steady-state assumption (or pseudo-steady-state hypothesis), namely that the concentration of the substrate-bound enzyme ([ES]) changes much more slowly than those of the product ([P]) and substrate ([S]). This allows us to set the rate of change of [ES] to zero and also write down the rate of product formation :*

$$\frac{d[ES]}{dt} = k_1[E][S] - [ES](k_{-1} + k_2) \stackrel{!}{=} 0 \qquad (2)$$

$$\frac{d[P]}{dt} = k_2[ES] \qquad (3)$$

En application de ce raisonnement, une idée très naturelle consiste à ajouter l'équation (2) ci-dessus au modèle déterministe naturel et à appliquer notre algorithme de simplification. Voici ce que cela donne.

```
violent_approx := [diff (ES(t), t) = 0, op (natsys)];
```

$$violent\_approx := \left[\frac{d}{dt} ES(t) = 0,\ \frac{d}{dt} E(t) = -k_1 E(t) S(t) + k_{-1} ES(t) + k_2 ES(t),\ \frac{d}{dt} S(t) = -k_1 E(t) S(t) + k_{-1} ES(t),\ \frac{d}{dt} ES(t) = k_1 E(t) S(t) - k_{-1} ES(t) - k_2 ES(t),\ \frac{d}{dt} P(t) = k_2 ES(t)\right]$$

L'algorithme de simplification retourne sept cas différents.

```
vapideal := RosenfeldGroebner (violent_approx, R);
```

$$vapideal := [regular\_differential\_chain, regular\_differential\_chain, regular\_differential\_chain, regular\_differential\_chain, regular\_differential\_chain, regular\_differential\_chain, regular\_differential\_chain]$$

En inspectant les équations des sept cas, on s'aperçoit que de nombreux cas sont des cas dégénérés, correspondant à l'annulation éventuelle des constantes cinétiques.

```
Equations (vapideal, solved);
```

$$\left[\left[\frac{d}{dt} S(t) = 0,\ \frac{d}{dt} ES(t) = 0,\ \frac{d}{dt} P(t) = 0,\ E(t) = \frac{ES(t)\, k_{-1}}{S(t)\, k_1},\ k_2 = 0\right],\ \left[\frac{d}{dt} S(t) = 0,\ \frac{d}{dt} P(t) = 0,\ E(t) = 0,\ ES(t) = 0\right],\ \left[\frac{d}{dt} S(t) = -k_2 ES(t),\ \frac{d}{dt} ES(t) = 0,\ \frac{d}{dt} P(t) = k_2 ES(t),\ E(t) = 0,\ k_{-1} = -k_2\right],\ \left[\frac{d}{dt} E(t) = 0,\ \frac{d}{dt} S(t) = -k_2 ES(t),\ \frac{d}{dt} ES(t) = 0,\ \frac{d}{dt} P(t) = k_2 ES(t),\ k_1 = 0,\ k_{-1} = -k_2\right],\ \left[\frac{d}{dt} E(t) = 0,\ \frac{d}{dt} S(t) = 0,\ \frac{d}{dt} P(t) = 0,\ ES(t) = 0,\ k_1 = 0\right],\ \left[\frac{d}{dt} E(t) = 0,\ \frac{d}{dt} ES(t) = 0,\ \frac{d}{dt} P(t) = 0,\ S(t) = 0,\ k_{-1} = 0,\ k_2 = 0\right],\ \left[\frac{d}{dt} E(t) = 0,\ \frac{d}{dt} P(t) = 0,\ S(t) = 0,\ ES(t) = 0\right]\right]$$

Pour y voir plus clair, on souhaiterait éviter ces cas dégénérés. Il y a plusieurs façons de s'y prendre avec le paquetage. La méthode présentée ci-dessous consiste à demander à *RosenfeldGroebner* de considérer que les trois constantes cinétiques font partie du corps des coefficients des équations. Les calculs sont alors effectués au-dessus du corps de fractions rationnelles $\mathbb{Q}(k_1, k_{-1}, k_2)$. Dans un corps, tout élément non nul est inversible. Les trois constantes cinétiques, étant des éléments non nuls du corps, ne peuvent pas s'annuler et certaines discussions de cas sont évitées.

```
vapideal := RosenfeldGroebner
              (violent_approx, R,
               basefield = field (generators = [k[1],k[-1],k[2]]));
```

$$vapideal := [regular\_differential\_chain, regular\_differential\_chain]$$

Il ne reste plus que deux cas : soit $E(t) = 0$ et $ES(t) = 0$, soit $S(t) = 0$ et $ES(t) = 0$. En d'autres termes, la réaction réversible ne se déclenche pas du tout.

```
Equations (vapideal, solved);
```

$$\left[\left[\frac{d}{dt}S(t)=0,\ \frac{d}{dt}P(t)=0,\ E(t)=0,\ ES(t)=0\right],\left[\frac{d}{dt}E(t)=0,\ \frac{d}{dt}P(t)=0,\ S(t)=0,\ ES(t)=0\right]\right]$$

Manifestement l'hypothèse énoncée en début de section est trop violente puisqu'elle conduit, si on en tire toutes les conséquences, comme le fait *RosenfeldGroebner*, à des dynamiques trop simplifiées.

4 Une meilleure approximation

L'idée consiste à supposer que les deux réactions à gauche du système sont rapides par rapport à la troisième, c'est-à-dire, en quelque sorte, que $k_1,\ k_{-1} \gg k_2$. Toute l'astuce consiste à coder cette idée par un système d'équations différentielles algébriques. En Anglais, une DAE. La méthode exposée ici a été énoncée pour la première fois dans [6]. L'idée est essentiellement due à F. Lemaire.

La première étape consiste à oublier, dans le modèle déterministe naturel, les formes précises $k_1\,E(t)\,S(t)$ et $k_{-1}\,ES(t)$ des contributions des deux réactions rapides, et à les remplacer par des contributions inconnues[1] $F_1(t)$ et $F_{-1}(t)$. Ce codage oublie toutes les informations concernant les réactions rapides, sauf une : à chaque fois qu'un réactant est consommé, un produit apparaît. En d'autres termes, les deux réactions rapides sont ... des réactions !

```
newV := <F[1](t), F[2](t), k[2]*ES(t)>:
V, newV;
```

$$\begin{bmatrix} k_1\,E(t)\,S(t) \\ k_{-1}\,ES(t) \\ k_2\,ES(t) \end{bmatrix}, \begin{bmatrix} F_1(t) \\ F_{-1}(t) \\ k_2\,ES(t) \end{bmatrix}$$

```
newsys0 := map (diff, X, t) = N . newV;
```

$$newsys0 := \begin{bmatrix} \frac{d}{dt}E(t) \\ \frac{d}{dt}S(t) \\ \frac{d}{dt}ES(t) \\ \frac{d}{dt}P(t) \end{bmatrix} = \begin{bmatrix} -F_1(t)+F_{-1}(t)+k_2\,ES(t) \\ -F_1(t)+F_{-1}(t) \\ F_1(t)-F_{-1}(t)-k_2\,ES(t) \\ k_2\,ES(t) \end{bmatrix}$$

[1] On peut aussi, traiter les deux réactions comme une seule réaction, et remplacer la contribution $k_1\,E(t)\,S(t) - k_{-1}\,ES(t)$ par une seule contribution inconnue (au lieu de deux) $F(t)$.

La seconde étape consiste à rajouter l'équation algébrique $k_1\,E(t)\,S(t) - k_{-1}\,ES(t) = 0$ au système différentiel. Cette équation s'obtient en oubliant les réactions lentes, et en posant que les réactions rapides sont à l'équilibre. Il ne reste plus qu'à simplifier le système d'équations différentielles algébriques ainsi obtenu.

```
newsys :=
    [ seq (lhs (newsys0) [i] = rhs (newsys0) [i],
           i = 1 .. Dimension (X)),
      0 = k[1]*E(t)*S(t) - k[-1]*ES(t) ];
```

$$newsys := \left[\frac{d}{dt} E(t) = -F_1(t) + F_{-1}(t) + k_2\,ES(t),\ \frac{d}{dt} S(t) = -F_1(t) + F_{-1}(t),\ \frac{d}{dt} ES(t) = F_1(t) - F_{-1}(t) - k_2\,ES(t),\ \frac{d}{dt} P(t) = k_2\,ES(t),\ 0 = k_1\,E(t)\,S(t) - k_{-1}\,ES(t) \right]$$

Reste à simplifier le système d'équations différentielles algébriques. Comme la formule de Henri, Michaelis et Menten, décrit l'évolution de la concentration du substrat $S(t)$ en fonction d'elle-même, on place le symbole S le plus à droite possible dans le bloc des quatre concentrations. Quant aux deux contributions inconnues, qu'on souhaite éliminer, on les place dans un bloc séparé, le plus à gauche possible.

```
R := DifferentialRing
        (blocks = [[F[1],F[-1]],
                   [E,ES,P,S],
                   [k[1](),k[-1](),k[2]()]],
         derivations = [t]);
```

$$R := differential\_ring$$

Pour limiter les discussions de cas, on demande que les trois constantes cinétiques soient considérées comme des éléments du corps des coefficients des équations.

```
newideal := RosenfeldGroebner
               (newsys, R,
                basefield = field (generators = [k[1],k[-1],k[2]]));
```

$$newideal := [regular\_differential\_chain,\ regular\_differential\_chain,\ regular\_differential\_chain]$$

La simplification conduit à trois cas. Les deux derniers sont manifestement des cas dégénérés.

```
Equations (newideal, solved);
```

$$\left[\left[F_1(t) = -\frac{-F_{-1}(t)\,ES(t)\,k_{-1} - F_{-1}(t)\,S(t)^2\,k_1 - F_{-1}(t)\,S(t)\,k_{-1} - ES(t)\,S(t)^2\,k_1\,k_2 - ES(t)\,S(t)\,k_{-1}\,k_2}{k_{-1}\,ES(t) + S(t)^2\,k_1 + S(t)\,k_{-1}},\ \frac{d}{dt} ES(t) = -\frac{ES(t)^2\,k_{-1}\,k_2}{k_{-1}\,ES(t) + S(t)^2\,k_1 + S(t)\,k_{-1}},\ \frac{d}{dt} P(t) = k_2\,ES(t),\ \frac{d}{dt} S(t) = -\frac{ES(t)\,S(t)^2\,k_1\,k_2 + ES(t)\,S(t)\,k_{-1}\,k_2}{k_{-1}\,ES(t) + S(t)^2\,k_1 + S(t)\,k_{-1}},\ E(t) = \frac{ES(t)\,k_{-1}}{S(t)\,k_1} \right], \left[F_1(t) = F_{-1}(t),\ \frac{d}{dt} P(t) = 0,\ E(t) \right]$$

$$= 0,\ ES(t) = 0,\ S(t) = -\frac{k_{-1}}{k_1}\Big],\ \Big[F_1(t) = F_{-1}(t),\ \frac{d}{dt}E(t) = 0,\ \frac{d}{dt}P(t) = 0,\ ES(t) = 0,\ S(t) = 0\Big]\Big]$$

La formule recherchée est la suivante. Ce n'est pas encore la formule de Michaelis-Menten, parce que toutes les hypothèses n'ont pas encore été prises en compte.

```
Equations (newideal[1], solved, leader = diff(S(t),t));
```

$$\Big[\frac{d}{dt}S(t) = -\frac{ES(t)\ S(t)^2\ k_1\ k_2 + ES(t)\ S(t)\ k_{-1}\ k_2}{k_{-1}\ ES(t) + S(t)^2\ k_1 + S(t)\ k_{-1}}\Big]$$

5 Prise en compte des hypothèses mineures

L'idée consiste à tenir compte des lois de conservation linéaires du système de réactions chimiques, d'hypothèses sur les conditions initiales ($P(0) = ES(0) = 0$ et $S(0) \gg E(0)$) et de renommer les constantes. On commence par définir un nouvel anneau de polynômes différentiels, en rajoutant quatre nouvelles constantes pour désigner les concentrations initiales, ainsi que les deux constantes K et $V_{\max}$ de la formule.

```
R := DifferentialRing
        (derivations = [t],
         blocks = [[F[1],F[-1]],
                   [E,ES,P,S],
                   [k[1](),k[-1](),k[2]()],
                   [E0(),S0(),ES0(),P0()],
                   [K(),Vmax()]]);
```

$$R := differential\_ring$$

5.1 Les lois de conservation linéaires

Les lois de conservation linéaires peuvent s'obtenir par une simple élimination de Gauss sur la transposée de la matrice de stœchiométrie N. Par exemple, si on additionne la première ligne (la ligne de E) et la troisième ligne (la ligne de ES) de N, on obtient une ligne nulle. Cela implique que $E(t) + ES(t) = cste = E(0) + ES(0)$.

```
conservation_laws :=
     [E(t) + ES(t) = E0 + ES0,
      S(t) + ES(t) + P(t) = S0 + ES0 + P0];
```

$$conservation\_laws := [E(t) + ES(t) = E0 + ES0,\ S(t) + ES(t) + P(t) = S0 + ES0 + P0]$$

5.2 Les conditions initiales nulles

```
zero_iv := [P0 = 0, ES0 = 0];
```

$$zero\_iv := [P0 = 0,\ ES0 = 0]$$

5.3 Le renommage des constantes

Les deux équations suivantes ne servent qu'à renommer les constantes. Pour que le renommage se fasse dans le bon sens, il faut que les constantes qu'on souhaite voir apparaître dans les équations (les constantes K et $V_{\max}$) fassent partie d'un bloc plus à droite que les constantes k et E_0 qu'on souhaite renommer (et donc éliminer). C'est effectivement le cas.

```
renaming := [K = k[-1]/k[1], Vmax = k[2]*E0];
```

$$renaming := \left[K = \frac{k_{-1}}{k_1},\ Vmax = k_2\ E0\right]$$

5.4 Le corps des coefficients

Pour éviter les cas dégénérés, on souhaite que la fonction *RosenfeldGroebner* traite toutes les constantes comme des éléments du corps de base des équations. Mais dans ce cas-ci, ces constantes (les générateurs du corps) ne sont pas indépendantes : elles satisfont des relations. Le paquetage *DifferentialAlgebra* permet de définir de tels corps, présentés par générateurs et relations. Les relations doivent alors former une chaîne différentielle régulière[2].

```
fieldrels := Tools:-PretendRegularDifferentialChain
                      ([op(zero_iv), op(renaming)], R);
```

$$fieldrels := regular\_differential\_chain$$

Voici les relations satisfaites par les générateurs du corps. On remarque que, dans les équations de renommage des constantes, les constantes qu'on souhaite voir apparaître, figurent maintenant en partie droite des équations.

```
Equations (fieldrels, solved);
```

$$\left[k_1 = \frac{k_{-1}}{K},\ k_2 = \frac{Vmax}{E0},\ ES0 = 0,\ P0 = 0\right]$$

5.5 La formule de Michaelis-Menten

Il suffit maintenant d'appliquer *RosenfeldGroebner* sur toutes les équations ensemble. La discussion se réduit à un seul cas.

```
ideal := RosenfeldGroebner
            ([ op(newsys), op(conservation_laws) ], R,
             basefield = field
                (generators = [E0,S0,ES0,P0,k[1],k[-1],k[2],K,Vmax],
                 relations = fieldrels));
```

[2]Et même une chaîne définissant un idéal différentiel premier, ce qui est le cas ici.

$$ideal := [regular\_differential\_chain]$$

Voici les équations de la chaîne différentielle régulière.

```
Equations (ideal[1], solved);
```

$$\left[F_1(t) = -\frac{-F_{-1}(t)\,S(t)^2 - 2\,F_{-1}(t)\,S(t)\,K - F_{-1}(t)\,E0\,K - F_{-1}(t)\,K^2 - S(t)^2\,Vmax - S(t)\,K\,Vmax}{S(t)^2 + 2\,S(t)\,K + E0\,K + K^2},\right.$$
$$\frac{d}{dt}S(t) = -\frac{S(t)^2\,Vmax + S(t)\,K\,Vmax}{S(t)^2 + 2\,S(t)\,K + E0\,K + K^2},\ E(t) = \frac{E0\,K}{S(t)+K},\ ES(t) = \frac{S(t)\,E0}{S(t)+K},\ P(t) =$$
$$\left.-\frac{S(t)^2 + S(t)\,E0 - S(t)\,S0 + S(t)\,K - S0\,K}{S(t)+K},\ k_1 = \frac{k_{-1}}{K},\ k_2 = \frac{Vmax}{E0},\ ES0 = 0,\ P0 = 0\right]$$

Voici la formule recherchée.

```
formula := Equations (ideal[1], leader = diff (S(t),t), solved)[1];
```

$$formula := \frac{d}{dt}S(t) = -\frac{S(t)^2\,Vmax + S(t)\,K\,Vmax}{S(t)^2 + 2\,S(t)\,K + E0\,K + K^2}$$

Enfin presque. Il reste à tenir compte de la dernière hypothèse $S(0) \gg E(0)$ et à négliger le terme $E_0\,K$ au dénominateur. A ma connaissance, il n'y a aucun moyen d'effectuer cette simplification par élimination différentielle !

```
map (normal, algsubs (K*E0 = 0, formula));
```

$$\frac{d}{dt}S(t) = -\frac{S(t)\,Vmax}{S(t)+K}$$

Quid de l'évolution de $P(t)$? La chaîne différentielle régulière donne directement la valeur de $P(t)$. Pour obtenir une équation différentielle pour le produit, il suffit de calculer la forme normale de la dérivée de $P(t)$ vis-à-vis de la chaîne différentielle régulière. On obtient la même formule que pour le substrat, mais sans utiliser l'hypothèse que $S(0) \gg E(0)$.

```
diff (P(t),t) = NormalForm (diff (P(t),t), ideal[1]);
```

$$\frac{d}{dt}P(t) = \frac{S(t)\,Vmax}{S(t)+K}$$

6 Le même calcul, sans MAPLE

Les calculs ci-dessus peuvent être effectués directement en utilisant BLAD, sans utiliser MAPLE. Voici le code d'un programme qui calcule la formule précédente et l'affiche à l'écran.

```
#include "bad.h"

int main ()
{   struct bad_base_field K;
    struct bad_regchain fieldrels;
    struct bad_intersectof_regchain T;
```

```
    struct bap_tableof_polynom_mpz eqns, ineqns, nulles;
    struct bap_ratfrac_mpz NF;
    struct bap_polynom_mpz poly;
    bav_Iordering r;

    bad_restart (0, 0);
    bav_set_settings_variable
                (&bav_scanf_diff_variable, &bav_printf_diff_variable, 0);
/*
 * Definition of the differential ring.
 */
    ba0_sscanf2 ("ordering (derivations = [t], \
                            blocks = [[F[1], F[-1]], [E, ES, P, S], \
                                      [k[1], k[-1], k[2]], \
                                      [E0, S0, ES0, P0], [K, Vmax]])",
                 "%ordering", &r);
    bav_R_push_ordering (r);
    ba0_sscanf2 ("[k[1], k[-1], k[2], E0, S0, ES0, P0, K, Vmax]",
                 "%t[%param]", &bav_parameters);
/*
 * The base field
 */
    bad_init_regchain (&fieldrels);
    ba0_sscanf2
        ("regchain ([P0, ES0, K - k[-1]/k[1], Vmax - k[2]*E0], \
                    [differential, squarefree, coherent, \
                     primitive, autoreduced, normalized, prime])",
         "%pretend_regchain", &fieldrels);
    bad_init_base_field (&K);
    bad_set_base_field_generators_and_relations
       (&K, (bav_tableof_variable)0, &fieldrels, &bav_parameters, true, false);
/*
 * The equations to be processed: the DAE plus the conservation laws
 */
    ba0_init_table ((ba0_table)&eqns);
    ba0_sscanf2 ("[diff(E(t),t)+F[1](t)-F[-1](t)-k[2]*ES(t), \
                   diff(S(t),t)+F[1](t)-F[-1](t), \
                   diff(ES(t),t)-F[1](t)+F[-1](t)+k[2]*ES(t), \
                   diff(P(t),t)-k[2]*ES(t), \
                   -k[1]*E(t)*S(t)+k[-1]*ES(t), \
                   E(t)+ES(t)-E0-ES0, \
                   S(t)+ES(t)+P(t)-S0-ES0-P0]", "%t[%Az]", &eqns);
/*
 * Plus the equations which express the fact that parameters are constant
 */
    ba0_init_table ((ba0_table)&nulles);
    bap_zero_derivatives_of_tableof_parameter_mpz (&nulles, &bav_parameters);
    ba0_concat_table ((ba0_table)&eqns, (ba0_table)&eqns, (ba0_table)&nulles);
/*
 * No inequation
 */
```

```
    ba0_init_table ((ba0_table)&ineqns);
    ba0_sscanf2 ("[]", "%t[%Az]", &ineqns);
/*
 * The elimination. Should lead to a single component
 */
    bad_init_intersectof_regchain (&T);
    ba0_sscanf2
        ("intersectof_regchain ([], \
                [differential, squarefree, coherent, primitive, \
                 autoreduced, normalized])",
         "%intersectof_regchain", &T);
    bad_Rosenfeld_Groebner (&T, &eqns, &ineqns, &K, (bad_splitting_control)0);
    bad_remove_redundant_components_intersectof_regchain (&T, &T);
/*
 * The normal form computation for diff (P(t), t)
 */
    bap_init_polynom_mpz (&poly);
    bap_init_ratfrac_mpz (&NF);
    ba0_sscanf2 ("diff(P(t),t)", "%Az", &poly);
    bad_normal_form_polynom_mod_regchain
                        (&NF, &poly, T.inter.tab [0], (bap_polynom_mpz*)0);
/*
 * Should print: diff (P(t),t) = (S(t)*Vmax)/(S(t) + K)
 */
    ba0_printf "The formula is: diff (P(t),t) = %Qz\n", &NF);
    bav_R_pull_ordering ();
    bad_terminate (ba0_init_level);
    return 0;
}
```

7 Compléments algébriques

Le texte qui suit est extrait d'un article de recherche. Il donne une description plus précise, mais moins pédagogique de la plupart des notions d'algèbre différentielle évoquées dans ce cours.

A *differential ring* R is a ring endowed with finitely many, say m, abstract *derivations* $\delta_1, \ldots, \delta_m$ i.e. unary operations which satisfy the following axioms :

$$\delta(a+b) = \delta(a) + \delta(b), \quad \delta(a\,b) = \delta(a)\,b + a\delta(b), \qquad (\forall\, a,\, b \in R)$$

and which are assumed to commute pairwise. This paper is mostly concerned by a differential polynomial ring R in n *differential indeterminates* $u_1, \ldots, u_n$ with coefficients in a commutative differential field K of characteristic zero, say $K = \mathbb{Q}$. Letting $U = \{u_1, \ldots, u_n\}$, one denotes $R = K\{U\}$, following Ritt and Kolchin. The set of derivations generates a commutative monoid w.r.t. the composition operation. It is denoted :

$$\Theta = \{\delta_1^{a_1} \cdots \delta_m^{a_m} \mid a_1, \ldots, a_m \in \mathbb{N}\}$$

where $\mathbb{N}$ stands for the set of the nonnegative integers. The elements of Θ are the *derivation operators*. If $\theta = \delta_1^{a_1} \cdots \delta_m^{a_m}$ is a derivation operator then $\operatorname{ord} \theta = a_1 + \cdots + a_m$ denotes its *order*. The monoid Θ acts on U, giving the infinite set ΘU of the *derivatives*. One indices derivations with letters e.g. δ_x, δ_y and one denotes derivatives using subscripts e.g. u_{xy} denotes $\delta_x \, \delta_y \, u$.

If A is a finite subset of R, one denotes (A) the smallest ideal containing A w.r.t. the inclusion relation and $[A]$ the smallest differential ideal containing A. Let $\mathfrak{A}$ be an ideal and $S = \{s_1, \ldots, s_t\}$ be a finite subset of R, not containing zero. Then

$$\mathfrak{A} : S^\infty = \{p \in R \mid \exists \, a_1, \ldots, a_t \in \mathbb{N}, \; s_1^{a_1} \cdots s_t^{a_t} \, p \in \mathfrak{A}\}$$

is called the *saturation* of $\mathfrak{A}$ by the multiplicative family generated by S. The saturation of a (differential) ideal is a (differential) ideal [13, chap. I, cor. to lem. 1].

Définition 1 *A* ranking *is a total ordering over ΘU which satisfies the two following axioms :*

1. $v \leq \theta v$ *for every* $v \in \Theta U$ *and* $\theta \in \Theta$,
2. $v < w \Rightarrow \theta v < \theta w$ *for every* $v, \, w \in \Theta U$ *and* $\theta \in \Theta$.

See [13, chap. I, sect. 8]. Rankings such that $\operatorname{ord} \theta < \operatorname{ord} \phi \Rightarrow \theta u < \phi v$ for every $\theta, \, \phi \in \Theta$ and $u, \, v \in U$ are called *orderly*.

Fix a ranking. Consider some differential polynomial $p \notin K$. The highest derivative v w.r.t. the ranking such that $\deg(p, v) > 0$ is called the *leading derivative* of p. It is denoted $\operatorname{ld} p$. The leading coefficient of p w.r.t. v is called the *initial* of p. The differential polynomial $\partial p / \partial v$ is called the *separant* of p. If C is a finite subset of $R \setminus K$ then I_C denotes its set of initials, S_C denotes its set of separants and $H_C = I_C \cup S_C$.

A differential polynomial q is said to be *partially reduced* w.r.t. p if it does not depend on any proper derivative of the leading derivative v of p. It is said to be *reduced* w.r.t. p if it is partially reduced w.r.t. p and $\deg(q, v) < \deg(p, v)$. A set of differential polynomials of $R \setminus K$ is said to be *autoreduced* if its elements are pairwise reduced. Autoreduced sets are necessarily finite [13, chap. I, sect. 9]. To each autoreduced set C, one may associate the set $L = \operatorname{ld} C$ of the leading derivatives of C and the set $N = \Theta U \setminus \Theta L$ of the derivatives which are not derivatives of any element of L (the derivatives "under the stairs" defined by C).

Ritt's reduction algorithm is a generalization of the classical *pseudoremainder* algorithm defined in [11, vol. 2, page 407], to differential polynomials. Ritt's algorithm is presented in [13, chap. I, sect. 9]. Given a differential polynomial p and an autoreduced set C, it permits to compute a set of exponents $a_1, \ldots, a_t \in \mathbb{N}$ and a differential polynomial p' partially reduced w.r.t. C (i.e. w.r.t. each element of C) such that $s_1^{a_1} \cdots s_t^{a_t} \, p \equiv p' \mod [C]$ where $s_1, \ldots, s_t$ denote the separants of the elements of C. Given a differential polynomial p' partially reduced w.r.t. C, it permits to compute a set of exponents $b_1, \ldots, b_t \in \mathbb{N}$ and a differential polynomial p'' reduced w.r.t. C such that $i_1^{b_1} \cdots i_t^{b_t} \, p' \equiv p'' \mod (C)$ where $i_1, \ldots, i_t$ denote the initials of the elements of C. When $p'' = 0$ one says that p' *is reduced to zero by* C.

Consider a set $C = \{c_1, \ldots, c_t\}$ of R and denote $\mathfrak{A} = [C] : H_C^\infty$. The following definition provides a compact presentation of regular differential chains, which were introduced by [15,

déf. 5]. Roughly speaking they are finite sets of differential polynomials satisfying both the regular chain condition, introduced by [1], and the hypotheses of [20, Lemma]. Very close concepts were introduced by [5] and [10].

Définition 2 *The set C is a* regular differential chain *if it satisfies the following conditions :*

a *the elements of C are pairwise partially reduced and have distinct leading derivatives ;*

b *for each $2 \leq k \leq t$, the initial i_k of c_k is regular in $K[N, L]/(c_1, \ldots, c_{k-1}) : (i_1 \cdots i_{k-1})^\infty$;*

c *for each $1 \leq k \leq t$, the separant s_k of c_k is regular in $K[N, L]/(c_1, \ldots, c_k) : (i_1 \cdots i_k)^\infty$;*

d *for any pair $\{c_k,\ c_\ell\}$ of elements of C, whose leading derivatives $\theta_k u$ and $\theta_\ell u$ are derivatives of some same differential indeterminate u, the Δ-polynomial*

$$\Delta(c_k,\ c_\ell) = s_\ell \, \frac{\theta_{k\ell}}{\theta_k} \, c_k - s_k \, \frac{\theta_{k\ell}}{\theta_\ell} \, c_\ell \, ,$$

where $\theta_{k\ell}$ denotes the least common multiple of θ_k and θ_ℓ, is reduced to zero by C, using Ritt's reduction algorithm.

Consider a regular differential chain C. Condition **a** is the *differentially triangular* condition of [5, def. 3]. It implies that C is triangular, algebraically. Algebraic triangularity plus condition **b** is equivalent to the *regular chain* condition of [1]. Condition **c** is then equivalent to the *squarefree regular chain* condition of [1]. See also [7]. Last, condition **d** is the *coherence* condition, which is the key condition of [20, Lemma]. See [5] and [10] for almost equivalent notions.

Définition 3 *Let f be a nonzero differential polynomial of R. An* inverse *of f is any fraction p/q of nonzero differential polynomials such that $p \in K[N \cup L]$ and $q \in K[N]$ and $f\, p \equiv q \mod \mathfrak{A}$.*

Définition 4 *Let a/b be a rational differential fraction, with b regular modulo $\mathfrak{A}$. A* normal form *of a/b modulo C is any rational differential fraction f/g such that*

1 *f is reduced with respect to C ;*

2 *g belongs to $K[N]$ (and is thus regular modulo $\mathfrak{A}$),*

3 *a/b and f/g are equivalent modulo $\mathfrak{A}$.*

Proposition 1 *Let a/b be a rational differential fraction, with b regular modulo $\mathfrak{A}$. The normal form f/g of a/b exists and is unique. In particular,*

4 *a belongs to $\mathfrak{A}$ if and only if its normal form is zero ;*

5 *f/g is a canonical representative of the residue class of a/b in the total fraction ring of $R/\mathfrak{A}$.*

Moreover,

6 *each irreducible factor of g divides the denominator of an inverse of b, or of some initial or separant of C .*

Conclusion

On a montré sur l'exemple de la réduction de Henri, Michaelis et Menten, que l'élimination différentielle permet d'effectuer une réduction du modèle déterministe naturel d'un système de réactions chimiques, sous l'hypothèse que les réactions sont réparties en deux groupes : les lentes et les rapides.

La méthode est générale.

Elle correspond en fait à un type de réduction connu, qu'on retrouve plus ou moins dans [24, 25, 2]. La formulation par l'algèbre différentielle présente l'avantage d'être complètement algorithmique et de masquer une étape difficile, qui consiste à inverser une matrice jacobienne dont les entrées sont des symboles liés par des relations.

Comme dans le cas de l'exemple traité, il est souvent utile de renommer les constantes, ou, plus généralement, de reparamétriser le système après élimination. Cette reparamétrisation peut s'algorithmiser par des calculs de symétries d'équations différentielles, qui généralisent un peu les techniques d'adimensionnement.

Un prototype de logiciel complet (paquetage MABSys) a été réalisé dans [23]. Il s'appuie sur le paquetage [14, *RegularChains*] pour la simplification des DAE et sur le paquetage [21, *ExpandedLiePointSymmetry*] pour la reparamétrisation.

Ce logiciel a permis de réduire le modèle d'un gène régulé par un polymère de sa propre protéine. Le modèle a été suffisamment réduit pour caractériser les valeurs des paramètres conduisant à un comportement oscillant [23]. Cette caractérisation n'a pas pu être réalisée sur le modèle déterministe naturel.

Références

[1] Philippe Aubry, Daniel Lazard, and Marc Moreno Maza. On the theories of triangular sets. *Journal of Symbolic Computation*, 28 :105–124, 1999.

[2] Matthew R. Bennet, Dmitri Volfson, Lev Tsimring, and Jeff Hasty. Transient Dynamics of Genetic Regulatory Networks. *Biophysical Journal*, 92 :3501–3512, May 2007.

[3] François Boulier. The BLAD libraries. `http://www.lifl.fr/~boulier/BLAD`, 2004.

[4] François Boulier, Daniel Lazard, François Ollivier, and Michel Petitot. Representation for the radical of a finitely generated differential ideal. In *ISSAC'95 : Proceedings of the 1995 international symposium on Symbolic and algebraic computation*, pages 158–166, New York, NY, USA, 1995. ACM Press. `http://hal.archives-ouvertes.fr/hal-00138020`.

[5] François Boulier, Daniel Lazard, François Ollivier, and Michel Petitot. Computing representations for radicals of finitely generated differential ideals. *Journal of AAECC*, 20(1) :73–121, 2009. (1997 Techrep. IT306 of the LIFL).

[6] François Boulier, Marc Lefranc, François Lemaire, and Pierre-Emmanuel Morant. Model Reduction of Chemical Reaction Systems using Elimination, 2007. Presented at the international conference MACIS 2007, `http://hal.archives-ouvertes.fr/hal-00184558`, submitted to Mathematics in Computer Science.

[7] François Boulier, François Lemaire, and Marc Moreno Maza. Well known theorems on triangular systems and the D^5 principle. In *Proceedings of Transgressive Computing 2006*, pages 79–91, Granada, Spain, 2006. http://hal.archives-ouvertes.fr/hal-00137158.

[8] Victor Henri. *Lois générales de l'Action des Diastases*. Hermann, Paris, 1903.

[9] F. Horn and R. Jackson. General mass action kinetics. *Archive for Rational Mechanics and Analysis*, 47 :81–116, 1972.

[10] Évelyne Hubert. Factorization free decomposition algorithms in differential algebra. *Journal of Symbolic Computation*, 29(4,5) :641–662, 2000.

[11] Donald Erwin Knuth. *The art of computer programming*. Addison–Wesley, 1966. Second edition.

[12] Petar Kokotovic, Hassan K. Khalil, and John O'Reilly. *Singular Perturbation Methods in Control : Analysis and Design*. Classics in Applied Mathematics 25. SIAM, 1999.

[13] Ellis Robert Kolchin. *Differential Algebra and Algebraic Groups*. Academic Press, New York, 1973.

[14] François Lemaire, Marc Moreno Maza, and Yuzhen Xie. The RegularChains library in MAPLE 10. In Ilias S. Kotsireas, editor, *The MAPLE conference*, pages 355–368, 2005.

[15] François Lemaire. *Contribution à l'algorithmique en algèbre différentielle*. PhD thesis, Université Lille I, 59655, Villeneuve d'Ascq, France, January 2002. (in French).

[16] Leonor Michaelis and Maud Menten. Die kinetik der invertinwirkung. *Biochemische Zeitschrift*, 49 :333–369, 1973. Partial translation in english on http://web.lemoyne.edu/~giunta/menten.html.

[17] Péter Érdi and János Tóth. *Mathematical models of chemical reactions : theory and applications of deterministic and stochastic models*. Princeton University Press, 1989.

[18] Joseph Fels Ritt. *Differential equations from the algebraic standpoint*, volume 14 of *American Mathematical Society Colloquium Publications*. AMS, New York, 1932.

[19] Joseph Fels Ritt. *Differential Algebra*. Dover Publications Inc., New York, 1950.

[20] Azriel Rosenfeld. Specializations in differential algebra. *Trans. Amer. Math. Soc.*, 90 :394–407, 1959.

[21] Alexandre 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.

[22] Abraham Seidenberg. An elimination theory for differential algebra. *Univ. California Publ. Math. (New Series)*, 3 :31–65, 1956.

[23] Aslı Ürgüplü. *Contribution to Symbolic Effective Qualitative Analysis of Dynamical Systems ; Application to Biochemical Reaction Networks*. PhD thesis, University Lille I, Lille, France, 2010.

[24] Vincent Van Breusegem and George Bastin. Reduced order dynamical modelling of reaction systems : a singular perturbation approach. In *Proceedings of the 30th IEEE Conference on Decision and Control*, pages 1049–1054, Brighton, England, December 1991.

[25] Nishith Vora and Prodromos Daoutidis. Nonlinear model reduction of chemical reaction systems. *AIChE Journal*, 47(10) :2320–2332, 2001.