HAL
open science

The Management of Parameters in the MAPLE DifferentialAlgebra Package

François Boulier

▶ To cite this version:

François Boulier. The Management of Parameters in the MAPLE DifferentialAlgebra Package. 2018. hal-01825191

HAL Id: hal-01825191
https://hal.science/hal-01825191v1

Preprint submitted on 28 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

The Management of Parameters in the MAPLE DifferentialAlgebra Package

François Boulier

June 28, 2018

This technical report contains a description of the management of the parameters in the MAPLE `DifferentialAlgebra` package and, in particular, in the `RosenfeldGroebner` function.

1 Need of Algebraic Specifications

The Rosenfeld-Groebner algorithm is an elimination method in differential algebra [11, 10]. Many different versions of it were written [3, 6, 2, 7] and implemented: a first version in the `diffalg` MAPLE package [5], a second version wihing the `BLAD` libraries [1], available in MAPLE through the `DifferentialAlgebra` package [4]. In this paper, we describe so far never published features of the `BLAD/DifferentialAlgebra` version, which are related to the management of parameters.

This software was designed with the long term idea that it could be used within a computational process which would involve differential elimination as an intermediate step. Such a situation would possibly occur within a software dedicated to the investigation of the properties of models presented or internally encoded by systems of nonlinear differential equations.

In such a situation, it is indeed essential for the output of the software to be specified *algebraically.* The argument is easy to understand over the more familiar Gaussian elimination: consider a square linear system $A\,x = b$ to be solved. If one uses the $L\,U$ decomposition of A then one gets an algebraic relation $A = L\,U$ between three matrices which permits to transform the problem "solving $A\,x = b$" into "solving $L\,y = b$ (first) and $U\,x = y$ (second)". If one uses the tacky Gaussian elimination, the former reduction problem gets difficult to explain, because of the lack of algebraic relation between the input matrix A and the elimination output.

2 The Different Types of Parameters

A polynomial differential system which encodes a mathematical model is likely to involve a large set of parameters (in models all constants are denoted by letters: masses are denoted m, lengthes ℓ ...) and, during the simplification process undertaken by the Rosenfeld-Groebner algorithm, the right way to manage these parameters may depend on the parameter. We may distinguish the following different types of situation:

A this type corresponds to parameters numerically known though denoted by a letter for legibility (the constant g of Newton $F = m\,g$ law);

B this type corresponds to parameters which are not supposed to be perfectly defined (in biological modeling or in epidemiology, models do not follow from physical laws and are rough approximations of the real phenomenon). It thus hardly makes sense to assume that these parameters satisfy any specific relation;

C this type corresponds to parameters constrained by algebraic relations which are known in advance (the three edges of a rectangle triangle);

D this type corresponds to parameters constrained by algebraic relations which are not known in advance. One may then possibly want to discuss the result of the elimination process by considering the vanishing or non vanishing of various algebraic relations arising during computations.

3 Mathematical Handling of the Different Types

From a computational point of view, the management of parameters essentially amounts to control the way the Rosenfeld-Groebner algorithm splits cases. From an algebraic point of view, this case splitting mechanism is controlled by acting on the base field of the polynomial ring.

This is an important point of the design of the `DifferentialAlgebra` package and a controversial one! Indeed, in algebraic text books, polynomials are always considered as elements of a single, well-defined polynomial ring. One is not supposed to change the ring to which a polynomial belongs. Switching a polynomial from one ring to another is actually possible but one then needs to define a ring homomorphism and consider the image of the polynomial through this homorphism. In constrast, in applied text books — or in computer algebra software such as MAPLE — equations arise first and the authors (who may be brilliant scientists) may not even bother to define the algebraic structure these equations belong to. The `DifferentialAlgebra` package approach is somewhere in between: the equations may be entered first; the polynomial ring they belong to is specified at simplification time as an argument to the `RosenfeldGroebner` function. The result of this design is 1) soundness: a polynomial or an ideal always belongs to a precise ring when a computation is undertaken; 2) flexibility: the ring need not be uniquely defined[1].

Let us thus assume that we want to process a polynomial differential system with numerical coefficients in the field of the rational numbers and depending on a single differential indeterminate y and a single parameter θ. Assume that we have a single derivation δ acting on the differential indeterminate, yielding derivatives. Introduce a symbol x such that $\delta\, x = 1$ so that the differential indeterminate can be viewed as an unknown function $y(x)$ and the derivation as $\mathrm{d}/\mathrm{d}x$. The "single" assumption if for legibility only.

For types **A** and **B**, the differential polynomial ring of the computation is

$$K\{y\} \quad \text{where} \quad K = \mathrm{Fr}(\mathbb{Q}\{\theta, x\}/\{\delta\theta,\, \delta x - 1\})\,.$$

With words, the base field K is the field of fractions of the differential polynomial ring $\mathbb{Q}\{\theta, x\}$ quotiented by the prime differential ideal $\{\delta\theta,\, \delta x - 1\}$. In the ring $K\{y\}$, every nonzero algebraic relation constraining θ is a nonzero element c of the field K. It is thus invertible and the equation $c = 0$ is equivalent to $1 = 0$. The Rosenfeld-Groebner algorithm will thus discard any case leading to such a relation.

For type **C**, let us denote $\mathfrak{t}$ the ideal of $\mathbb{Q}[\theta]$ of the known relations which constrain the parameter θ (in the case of a single parameter, $\mathfrak{t}$ is a principal ideal but this needs not be true in the general case). Then, the differential polynomial ring of the computation is

$$K\{y\} \quad \text{where} \quad K = \mathrm{Fr}(\mathbb{Q}\{\theta, x\}/\{\mathfrak{t},\, \delta\theta,\, \delta x - 1\})\,.$$

Observe that, if t is any polynomial of $\mathfrak{t}$ then δt is a linear combination of monomials which all admit $\delta\theta$ as a factor. Thus $\{\mathfrak{t},\, \delta\theta\} \cap \mathbb{Q}[\theta]$ is equal to the radical of $\mathfrak{t}$. Assume $\mathfrak{t}$ is prime (this is the usual case). Then $\{\mathfrak{t},\, \delta\theta,\, \delta x - 1\}$ is so and K is a field (dropping this assumption, K would be isomorphic to a direct product of fields — one field per isolated associated prime ideal of $\mathfrak{t}$). In the ring $K\{y\}$, an algebraic relation constraining θ is either zero (if it belongs to $\{\mathfrak{t},\, \delta\theta\}$) or invertible (if it does not). The Rosenfeld-Groebner algorithm will thus discard any case leading to a relation which does not belong to $\{\mathfrak{t},\, \delta\theta\}$.

For type **D**, the differential polynomial ring of the computation is

$$R\{y\} \quad \text{where} \quad R = \mathrm{Fr}(\mathbb{Q}\{x\}/\{\delta x - 1\})\{\theta\}/\{\delta\theta\}\,.$$

[1] Let us stress that we do not think that the algebraic text books presentation implies that a polynomial belongs to a precised single ring. The authors (who may also be brilliant scientists) may just want to fix one of many different possible rings before investigating algebraic issues (the choice of the ring being out of the scope of the book). A somewhat similar situation arises in numerical computing: in numerical text books, algorithms are studied independently of any precise question and it is common to focus on relative errors exclusively; however, as soon as a precise question is investigated, absolute errors enter in consideration.

With words, the parameter θ is withdrawn from the base field. It is considered as a differential indeterminate (as y is) implicitly constrained by the relation $\delta\theta = 0$. The Rosenfeld-Groebner algorithm will thus allow any case leading to any relation in $\mathbb{Q}[\theta] \setminus \mathbb{Q}$.

4 Computational Handling of the Different Types

Thanks to the regular differential chains algorithmic theory, the Rosenfeld-Groebner algorithm does not rely on any Gröbner basis computation[2]. See [8] for recent developments on this theory.

Regular differential chains are particular cases of triangular sets. The implementation idea is the following. Assume that the Rosenfeld-Groebner algorithm processes some differential polynomial system Σ in one of the rings introduced in section 3. This algorithm actually computes a tree of quadruples $\langle A, D, P, S\rangle$ where A is both the set of the (somehow) already processed equations and a regular chain. In the basic case (no parameters), the algorithm starts with the root $\langle A, D, P, S\rangle = \langle \varnothing, \varnothing, \Sigma, \varnothing\rangle$ and stops on leaves of the form $\langle A, D, P, S\rangle = \langle A, \varnothing, \varnothing, \varnothing\rangle$ where A is a regular differential chain.

In the case with parameters, is is assumed that the relations needed to define the fields K or the ring R introduced in section 3 are presented as a regular differential chain A' (this actually is a restriction on the ideal $\mathfrak{t}$ of the type **C**). Then the algorithm starts with the root $\langle A, D, P, S\rangle = \langle A', \varnothing, \Sigma, \varnothing\rangle$ and stops on leaves of the form $\langle A, D, P, S\rangle = \langle A, \varnothing, \varnothing, \varnothing\rangle$ where A is a regular differential chain containing A' as a regular differential subchain (see [8, Proposition 38] for a related theorem). Here are a few remarks:

- all base field defining relations are thus handled by the general algorithms for regular differential chains: zero decision, regularity decision, normal forms methods;
- relations such as $\delta\theta = 0$ or $\delta x = 1$ are detected and passed as arguments to any algorithm (such as the differentiation algorithm of differential polynomials) which would otherwise waste computation time;
- depending on the types of parameters, some cases may not be generated;
- unless required, the equations of A' such as $\delta\theta = 0$ are not displayed by a `DifferentialAlgebra` function such as `Equations`;
- in the partial differential case, this mechanism is extended to handle functions depending on a restricted set of independent variables. For instance, assume two derivations δ_x and δ_t encoding partial derivatives with respect to x and t are defined; then a function $\theta(x)$ is handled as a differential indeterminate θ subject to $\delta_t\theta = 0$.

5 Academic Examples

In this paper, we only provide academic examples. A more interesting one for type **C** is described in [9] and shows how the Michaelis-Menten formula can be obtained by a differential elimination process. Relations among the parameters come from hypotheses on conservation laws and equations meant to rename constants.

5.1 Type D Example

Let us start by the simplest type **D**. The empty parentheses following θ in the definition of the differential polynomial ring indicate that this "differential indeterminate" does not depend on x i.e. is a constant. The differential indeterminate u can be viewed as an unknown function $u(x)$. The symbol `u[x]` denotes the derivative δu:

[2]This was not the case in the early versions of the algorithm — as the name of the algorithm indicates it.

```
> with (DifferentialAlgebra):
> p := theta*(theta-1)*(u[x]**2 - 4*u);
                                                  2
                     p := theta (theta - 1) (u[x]  - 4 u)

> R := DifferentialRing (derivations = [x], blocks = [u,theta()]);
                             R := differential_ring
```

The Rosenfeld-Groebner algorithm produces four regular differential chains:

```
> ideal := RosenfeldGroebner ([p], R);
ideal := [regular_differential_chain, regular_differential_chain,

    regular_differential_chain, regular_differential_chain]
```

The possible vanishing of constant factors of p is indeed discussed (polynomials in regular differential chains should be viewed as left hand sides of equations):

```
> Equations (ideal);
                                                 2
                  [[theta], [theta - 1], [u[x]  - 4 u], [u]]
```

In the two last regular differential chains, the equation $\delta\theta = 0$ is not displayed. We can get it if desired:

```
> Equations (ideal, fullset=true);
                                      2
        [[theta], [theta - 1], [u[x]  - 4 u, theta[x]], [u, theta[x]]]
```

5.2 Types A and B Example

Let us now illustrate types **A** and **B**. For this, one introduces K as a differential field extension of $\mathbb{Q}$ defined by one generator θ and no relation, apart the implicit one $\delta\theta = 0$. Observe that the `field` keyword is a placeholder. It will only be interpreted when passed as an argument to the Rosenfeld-Groebner algorithm:

```
> K := field (generators = [theta]);
                       K := field(generators = [theta])
```

The Rosenfeld-Groebner algorithm computes a regular differential chain decomposition of the differential ideal $\{p\}$ in a differential polynomial ring obtained by changing the base field of R. It actually discards the two cases leading to constraints on θ:

```
> ideal := RosenfeldGroebner ([p], R, basefield=K);
       ideal := [regular_differential_chain, regular_differential_chain]

> Equations (ideal);
                                     2
                             [[u[x]  - 4 u], [u]]
```

5.3 Type C Example

Let us finish with type **C**. For this, one transforms K into a differential field extension of $\mathbb{Q}$ defined by one generator θ and a prime differential ideal of relations. This prime differential ideal is presented by a regular differential chain A':

```
> with (Tools):
> Aprime := PretendRegularDifferentialChain ([theta-1],notation=jet,R);
                     Aprime := regular_differential_chain
> K := field (generators = [theta], relations = Aprime);
   K := field(generators = [theta], relations = regular_differential_chain)
```

The Rosenfeld-Groebner algorithm now discards any case leading to a constraint on θ different from the one specified in A':

```
> ideal := RosenfeldGroebner ([p], R, basefield=K);
       ideal := [regular_differential_chain, regular_differential_chain]

> Equations (ideal);
                          2
                  [[u[x]  - 4 u, theta - 1], [u, theta - 1]]
```

References

[1] François Boulier. The BLAD libraries. http://cristal.univ-lille.fr/~boulier/BLAD.

[2] François Boulier. Réécriture algébrique dans les systèmes d'équations différentielles polynomiales en vue d'applications dans les Sciences du Vivant, May 2006. Mémoire d'habilitation à diriger des recherches. Université Lille I, LIFL, 59655 Villeneuve d'Ascq, France. http://tel.archives-ouvertes.fr/tel-00137153.

[3] François Boulier. *Étude et implantation de quelques algorithmes en algèbre différentielle*. PhD thesis, Université Lille I, 59655, Villeneuve d'Ascq, France, 1994. http://tel.archives-ouvertes.fr/tel-00137866.

[4] François Boulier and Edgardo Cheb-Terrab. DifferentialAlgebra. Package of MapleSoft MAPLE standard library since MAPLE 14, 2008.

[5] François Boulier and Évelyne Hubert. diffalg. Package of MapleSoft MAPLE standard library from MAPLE V to MAPLE 13, 1996.

[6] 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.

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

[8] François Boulier, François Lemaire, Marc Moreno Maza, and Adrien Poteaux. An Equivalence Theorem For Regular Differential Chains. *Journal of Symbolic Computation*, pages 1–27, 2018. In press.

[9] François Boulier, François Lemaire, Michel Petitot, and Alexandre Sedoglavic. Chemical Reaction Systems, Computer Algebra and Systems Biology. In Vladimir Gerdt et al., editor, *Proceedings of Computer Algebra in Scientific Computing, LNCS 6885*, pages 73–87, Kassel, Germany, 2011. http://hal.archives-ouvertes.fr/hal-00603290.

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

[11] Joseph Fels Ritt. *Differential Algebra*, volume 33 of *American Mathematical Society Colloquium Publications*. American Mathematical Society, New York, 1950.