Investigating Generic Methods to Solve Hopf Bifurcation Problems in Algebraic Biology

Thomas Sturm[1] and Andreas Weber[2]

[1] Universität Passau, 94030 Passau, Germany
sturm@uni-passau.de
[2] Institut für Informatik II, Universität Bonn, Römerstr. 164, 53117 Bonn, Germany
weber@cs.uni-bonn.de

Abstract. Symbolic methods for investigating Hopf bifurcation problems of vector fields arising in the context of algebraic biology have recently obtained renewed attention. However, the symbolic investigations have not been fully algorithmic but required a sequence of symbolic computations intervened with ad hoc insights and decisions made by a human. In this paper we discuss the use of generic methods to reduce questions on the existence of Hopf bifurcations in parameterized polynomial vector fields to quantifier elimination problems over the reals combined with simplification techniques available in REDLOG. We can reconstruct most of the results given in the literature within a few seconds of computation time. As no tedious hand computations are involved we presume that the use of these generic methods will be a useful tool for investigating other examples.

1 Introduction

Symbolic methods to investigate Hopf bifurcation problems of vector fields arising in the context of algebraic biology have recently obtained renewed attention [1–3]. A major reason for this attention is the relationship between Hopf bifurcation fixed points and the occurrence of oscillations. Although this relationship is subtle—we refer to [1, 3] for nice introductions containing further references—the topic of investigating "oscillations" in systems for biologically or chemically relevant values of parameters is of such great interest that considerable work has been done to investigate various biological and chemical systems with respect to Hopf bifurcation fixed points, see e.g. [4–8] to mention only some.

There exist software packages such as AUTO[3] or XPPAUT[4], which locate Hopf bifurcations by means of numerical calculations. They allow one to evidence the existence of Hopf bifurcations. However, if one wants to prove that no Hopf bifurcation fixed point exists at all, then numeric methods fail in principle. Furthermore, if there are ranges of parameters for which there are Hopf bifurcations but these ranges are rather small, then numeric methods might not

[3] indy.cs.concordai.ca/auto/
[4] www.math.pitt.edu/∼bard/xpp/xpp.html

detect them—a possibility that is not only of theoretical interest in the context of algebraic biology and chemical reaction systems.

Whereas theoretically the problem is known to be decidable [9, 2, 3] the symbolic investigations carried out for specific parameterized polynomial vector fields arising from larger examples, e.g. the ones investigated in [1, 2], have not been fully algorithmic up to now but required a sequence of symbolic computation intervened with ad hoc insights and decisions made by a human, and sometimes of sophisticated coordinate transforms. The aim of this paper is to demonstrate how to proceed fully algorithmically for the examples discussed there.

For this we proceed as follows: We use the method described by El Kahoui and Weber [9, 10] to generate from the symbolic description of the respective ordinary differential equation a first-order formula in the language of ordered rings, where our domain is the real numbers. This is always possible if the vector field consists of polynomials in the variables and parameters.

If one suspects that there is no Hopf bifurcation fixed point or one just wants to assert that there is one, then one can apply quantifier elimination to the existential closure, i.e. all parameters are existentially quantified, of our generated formula. More generally, applying quantifier elimination to the original formula yields in principle a quantifier-free semi-algebraic description of the parameters for which Hopf bifurcation fixed points exist. In practice this latter variant does not finish within reasonable time at present. There is, however, an alternative, which provides at least one sample solution in the positive case; viz. extended quantifier elimination [11].

For the quantifier elimination we use REDLOG [12], which provides an automatic combination of virtual substitution methods [13] with partial CAD [14]. After noticing quite soon that most of our generated formulae were intractable by the regular elimination method, we developed a modified quantifier elimination procedure called *positive quantifier elimination*. This is based on the observation that in the considered examples, the variables as well as almost all of the parameters can be restricted to positive values. It has turned out that this knowledge greatly supports the elimination process. A current version of REDLOG including positive quantifier elimination is available for free download on the REDLOG website[5]. All computational examples discussed throughout this paper are contained in the online database REMIS there [15].

We can reconstruct the results of [1] in some seconds of computation time. Also the main example given in [2] could be handled in principle—without using the sophisticated coordinate transform used there.

2 Quantifier Elimination over the Reals

2.1 A Survey of Regular Quantifier Elimination

In order to summarize the basic idea of real quantifier elimination, we introduce first-order logic on top of polynomial equations and inequalities.

[5] www.redlog.eu

We consider multivariate polynomials $f(u, x)$ with rational coefficients, where $u = (u_1 \ldots, u_m)$ and $x = (x_1, \ldots, x_n)$. We call u *parameters* and we call x *variables*. Equations will be expressions of the form $f = 0$, inequalities are of the form $f \leq 0$, $f < 0$, $f \geq 0$, $f > 0$, or $f \neq 0$. Equations and inequalities are called *atomic formulae*. *Quantifier-free formulae* are Boolean combinations of atomic formulae by the logical operators "$\wedge$," "$\vee$," and "$\neg$." *Existential formulae* are of the form $\exists x_1 \ldots \exists x_n \psi(u, x)$, where ψ is a quantifier-free formula. Similarly, *universal formulae* are of the form $\forall x_1 \ldots \forall x_n \psi(u, x)$. A general (prenex) *first-order formula* has several alternating blocks of existential and universal quantifiers in front of a quantifier-free formula.

The real *quantifier elimination problem* can be phrased as follows: Given a formula φ, find a quantifier-free formula φ' such that both φ and φ' are equivalent in the domain of the real numbers. A procedure computing such a φ' from φ is called a real *quantifier elimination procedure*.

Quantifier elimination for an existential formula $\varphi(u) \equiv \exists x_1 \ldots \exists x_n \psi(u, x)$ has a straightforward geometric interpretation: Let

$$M = \{\, (u, x) \in \mathbb{R}^{m+n} \mid \psi(u, x) \,\},$$

and let $M' = \{\, u \in \mathbb{R}^m \mid \varphi(u) \,\}$. Then M' is the projection of M along the coordinate axes of the existentially quantified variables x onto the parameter space. Quantifier elimination yields a quantifier-free description of this projection.

Sets defined by first-order formulae are called *definable sets*. Sets defined by quantifier-free formulae are called *semi-algebraic sets*. Obviously, every semi-algebraic set is definable. The existence of a quantifier elimination procedure implies that, vice versa, every definable set is already semi-algebraic.

Obviously, quantifier elimination for existential formulae provides a test that determines the solvability of a parametric system of equations in dependence on the parameters. The procedure gives, however, no information on possible *solutions* of the input system. This point of view gives rise to the following generalization of quantifier elimination: Given an existential formula $\varphi \equiv \exists x_1 \ldots x_n \psi(u, x)$, we wish to compute a set

$$\Phi' = \{\, \bigl(\varphi'_k(u), \alpha_k(u)\bigr) \mid k \in K \,\}, \quad K \text{ finite},$$

which has the following properties:

1. The φ'_k are quantifier-free formulae. The α_k provide terms $t_1(u)$, ..., $t_n(u)$ corresponding to the eliminated variables x_1, ..., x_n.
2. Define φ' as $\bigvee_{k \in K} \varphi'_k$. Then φ' is equivalent to φ in the reals. In other words, φ' is obtained from φ by quantifier elimination.
3. Fix real values for the parameters u: if φ and hence φ' holds, then there is some φ'_k which holds. The corresponding *answer* α_k is a sample point, which when *virtually substituted* for x satisfies ψ.

The notion of virtual substitution refers to the fact that the terms $t_1(u), \ldots, t_n(u)$ possibly contain some non-standard symbols like quotients, root expressions, or infinitesimals, which can be substituted into quantifier-free formulae in such a way that they do not formally occur in the substitution result. An algorithm mapping φ to Φ' as described above is called an *extended* quantifier elimination procedure, or a quantifier elimination *with answer.*

2.2 Positive Quantifier Elimination

In the context of algebraic biology it is often known that *all* (or at least most) variables and parameters are positive. Let us assume first that all variables and parameters are positive. Such a global information greatly supports the quantifier elimination process by virtual substitution. We call the resulting procedure *positive quantifier elimination.* We do not go into technical details on this here but indicate a few major points, where positivity can be exploited. To start with, recall that the practical applicability of substitution methods depends crucially on efficient and powerful simplification of intermediate and final results [16]. These simplification methods can be considerably improved by positivity assumptions on all variables:

- There is an established simplification strategy which checks for terms that are *(strict) trivial squaresums* [16]. These are by definition sums of monomials with positive coefficients and even degree in all variables (and a positive absolute summand). Obviously, trivial squaresums are positive semi-definite, and strict trivial squaresums are positive definite. Thus corresponding atomic formulae can be evaluated to true or false or at least be simplified to equations or negated equations. On the assumption that all variables are positive, we need not check for positive degrees. For instance, we generally know that

  $$3x^2y^4 + 7x^2 \geq 0.$$

  With positive quantifier elimination, we may go further and even equivalently replace by "true" the atomic formula

  $$3xy + 7x > 0.$$

  Notice that technically we need only literally check for a minus sign in some canonical recursive or distributive representation of left hand side polynomials.
- The search for (generalized) trivial squaresums can be extended to testing irreducible factors of the occurring polynomials. For instance, in the positive case we can go

  $$ax^2 + axy^2 - bx^2 - by^2 \leq 0 \longleftrightarrow (ax - b)(x + y^2) \leq 0 \longleftrightarrow ax - b \leq 0.$$

  Since our positivity assumptions greatly increase our chance to discover such squaresums, it pays off in many cases to factorize more systematically. In order to optionally do so we have introduced into REDLOG a switch `rlsifaco` ("simplifier factorization with ordering relations"), which is going to play a role in the discussion of our computation examples later on.

Next, we illustrate two situations, where we can exploit positivity within the virtual substitution process itself:

- When virtually substituting into atomic formulae quotients with parametric denominators one must in general multiply with the square of the denominator in order to preserve signs. As an example consider

$$\bigl(ax - b > 0\bigr)\bigl[x /\!\!/ \tfrac{c}{y+z^2}\bigr] \equiv acy + acz^2 - by^2 - 2byz^2 - bz^4 > 0.$$

  With positive quantifier elimination one heuristically discovers non-negative denominators or at least non-negative factors based on strategies as sketched for simplification above. In our example, this allows to simply drop the then positive denominator:

$$\bigl(ax - b > 0\bigr)\bigl[x /\!\!/ \tfrac{c}{y+z^2}\bigr] \equiv ac - by - bz^2 > 0.$$

  At this point one possibly avoids subsequent degree violations for quantified variables in the denominator; in our example, z might be a quantified variable.
- There is a degree decreasing shift operation which essentially divides all exponents of quantified variables by the GCD of these exponents [17]. Doing so, one obviously has to take care of positivity, e.g., when switching between even and odd degrees. This operation can be simplified and slightly generalized on our positivity assumption. As a simple example consider the following transformation, where with positive quantifier elimination we need not add a condition $z \geq 0$:

$$\exists z(az^2 - bz^4 > 0 \land z^4 - 3z^2 < 0) \longleftrightarrow \exists z(az - bz^2 > 0 \land z^2 - 3z < 0).$$

For our set of input formulas discussed throughout this paper, it turns out that all optimizations discussed above become relevant at some point. For input problems from other fields of applications, positive quantifier elimination has not been systematically evaluated yet.

Notice that positive quantifier elimination as a concept is not at all restricted to virtual substitution methods. For instance, during the projection phase of partial CAD one could drop polynomials that are definite on positivity assumptions.

It is theoretically quite straightforward to equip positive quantifier elimination with an optional argument to pass a list of parameters and variables that are *not* known to be positive. This is, however, not yet implemented at present. We are now going to discuss how to alternatively deal with a parameter that is not known to be positive, say λ_1.

In applications, it might happen that the parameter occurs in a linear equation. In this special case, one can eliminate it by solving for λ_1 in one of the linear equations involving this parameter. Algorithmically, this strategy can be realized in a preprocessing step, e.g. the one realized by Brown and Groß [18], but is of course restricted to special cases.

One generally applicable procedure is to perform a case distinction on $\lambda_1 > 0$, $\lambda_1 < 0$, or $\lambda_1 = 0$. This results in three positive quantifier elimination runs, where in the two latter cases we substitute $\lambda_1 \leftarrow -\lambda_1$ and $\lambda_1 \leftarrow 0$, respectively. Iterating this procedure for other general parameters $\lambda_2, \dots, \lambda_n$ yields 3^n many case distinctions, so that the case distinction is only feasible for small n, i.e. there are only few parameters not known to be positive.

There is another theoretically interesting option: For quantified non-positive variables we observe that every real number is a difference of two positive real numbers and go $\lambda_1 \leftarrow \lambda_1 - \lambda_1'$, where λ_1' is quantified in the same way as λ_1.

Note that both our substitution techniques sketched above are completely algorithmic. They can easily be implemented on top of any implementation of positive quantifier elimination.

3 Computation Examples

3.1 Models of Genetic Circuits

The examples investigated in [1] consist of a family of ordinary differential equations of the following form:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}G(t) &= \vartheta(\gamma_0 - G(t) - G(t)P(t)^n),\\
\frac{\mathrm{d}}{\mathrm{d}t}P(t) &= n\alpha(\gamma_0 - G(t) - G(t)P(t)^n) + \delta(M(t) - P(t)),\\
\frac{\mathrm{d}}{\mathrm{d}t}M(t) &= \lambda_1 G(t) + \gamma_0\mu - M(t), \qquad (1)
\end{aligned}$$

where n is a natural number. We have renamed λ to λ_1 because `lambda` is obviously a bad choice for a variable name in Lisp systems. All variables and parameters except λ_1 are known to be positive. For a description of the model, from which this family of ordinary differential equations arise, we refer to [1].

Using a Maple library[6] containing the methods described in [9] we can generate the first order formulae stating the question on the existence of Hopf bifurcation fixed points (taking into account the known positivity conditions) by the Maple script in Figure 1.

Here we generate formulae up to $n = 10$ in a slight extension of the cases considered in [1]. There it is proved that no Hopf bifurcations exist for $n \leq 8$ and a Hopf bifurcations fixed point exists for $n = 9$.

In Figure 2 we present the generated first-order input formulae for $n = 2$ and $n = 9$. The system variables are renamed to `vv1`, `vv2`, `vv3` by the Maple library; as they occur in quantified form only, this renaming is of little concern. For better readability they are typeset as $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$. Apart from this, we literally give the formulae as they are output by Maple. They happen to contain some redundant parentheses.

As indicated in the introduction, we consider for now exclusively the existential closures $\exists\varphi_2, \dots, \exists\varphi_{10}$ of our formulae. Then quantifier elimination yields

[6] The current version of the library can be obtained at cg.cs.uni-bonn.de/project-pages/symbolicanalysis/.

```
eq1n := diff(G(t),t)=(theta*(gamma0-G(t)-G(t)*P(t)^n));
eq2n := diff(P(t),t)=n*alpha*(gamma0-G(t)-G(t)*P(t)^n)+delta*(M(t)-P(t));
eq3n := diff(M(t),t)=lambda1*G(t)+gamma0*mu-M(t);
fcns1 := {G(t),P(t),M(t)};
params1 := [theta,alpha,gamma0,mu,lambda1];
paramcondlist := [theta>0, gamma0>0, mu>0, delta>0, alpha>0];
funccondlist := [G(t)>0, P(t)>0, M(t)>0];
lown:= 0; upn :=10;
for nn from lown to upn do
   eq11 := eval(subs(n=nn,eq1n));
   eq21 := eval(subs(n=nn,eq2n));
   eq31 := eval(subs(n=nn,eq3n));
   DEHopfexistence({eq11,eq21,eq31},
      fcns1,params1,funccondlist,paramcondlist);
od;
```

Fig. 1. A Maple script for generating the first-order input formulae $\varphi_2, \ldots, \varphi_{10}$ for Section 3.1.

$$\begin{aligned}
\varphi_2 \equiv\ & \exists \mathbf{v}_1 \exists \mathbf{v}_2 \exists \mathbf{v}_3(((0 < \mathbf{v}_1 \wedge 0 < \mathbf{v}_3) \wedge 0 < \mathbf{v}_2) \wedge \\
& (((((((\vartheta(\gamma_0 - \mathbf{v}_1 - \mathbf{v}_1\mathbf{v}_3^2) = 0 \wedge \\
& \lambda_1\mathbf{v}_1 + \gamma_0\mu - \mathbf{v}_2 = 0) \wedge \\
& 2\alpha(\gamma_0 - \mathbf{v}_1 - \mathbf{v}_1\mathbf{v}_3^2) + \delta(\mathbf{v}_2 - \mathbf{v}_3) = 0) \wedge \\
& (0 < \vartheta\delta + \vartheta\mathbf{v}_3^2\delta + 2\lambda_1\vartheta\mathbf{v}_1\mathbf{v}_3\delta \wedge \\
& \vartheta\mathbf{v}_3^2 + 2\vartheta\delta + 8\vartheta\alpha\mathbf{v}_1\mathbf{v}_3 + 4\alpha\mathbf{v}_1\mathbf{v}_3\vartheta\delta + 4\alpha\mathbf{v}_1\mathbf{v}_3^3\vartheta\delta + 8\alpha\mathbf{v}_1\mathbf{v}_3\delta + \delta^2 + \\
& \quad \vartheta\delta^2 + 16\alpha^2\mathbf{v}_1^2\mathbf{v}_3^2 + \vartheta\mathbf{v}_3^2\delta^2 + 2\vartheta^2\mathbf{v}_3^2\delta + \vartheta^2\mathbf{v}_3^4\delta + \delta + \vartheta^2\delta + \vartheta^2 + \\
& \quad 2\vartheta^2\mathbf{v}_3^2 + \vartheta^2\mathbf{v}_3^4 + 4\alpha\mathbf{v}_1\mathbf{v}_3 + \vartheta + 2\vartheta\mathbf{v}_3^2\delta + 8\vartheta\mathbf{v}_3^3\alpha\mathbf{v}_1 - 2\lambda_1\vartheta\mathbf{v}_1\mathbf{v}_3\delta = 0)) \wedge \\
& 0 < \vartheta) \wedge 0 < \gamma_0) \wedge 0 < \mu) \wedge 0 < \delta \wedge 0 < \alpha)),
\end{aligned}$$

$$\begin{aligned}
\varphi_9 \equiv\ & \exists \mathbf{v}_2 \exists \mathbf{v}_1 \exists \mathbf{v}_3(((0 < \mathbf{v}_1 \wedge 0 < \mathbf{v}_3) \wedge 0 < \mathbf{v}_2) \wedge \\
& (((((((\vartheta(\gamma_0 - \mathbf{v}_1 - \mathbf{v}_1\mathbf{v}_3^9) = 0 \wedge \\
& \lambda_1\mathbf{v}_1 + \gamma_0\mu - \mathbf{v}_2 = 0) \wedge \\
& 9\alpha(\gamma_0 - \mathbf{v}_1 - \mathbf{v}_1\mathbf{v}_3^9) + \delta(\mathbf{v}_2 - \mathbf{v}_3) = 0) \wedge \\
& (0 < \vartheta\delta + \vartheta\mathbf{v}_3^9\delta + 9\lambda_1\vartheta\mathbf{v}_1\mathbf{v}_3^8\delta \wedge \\
& 162\vartheta\mathbf{v}_3^{17}\alpha\mathbf{v}_1 + 162\vartheta\alpha\mathbf{v}_1\mathbf{v}_3^8 + 162\alpha\mathbf{v}_1\mathbf{v}_3^8\delta + \vartheta + 2\vartheta\mathbf{v}_3^9\delta + \vartheta^2\mathbf{v}_3^{18}\delta + \\
& \quad \vartheta\mathbf{v}_3^9 + 2\vartheta\delta + 81\alpha\mathbf{v}_1\mathbf{v}_3^8\vartheta\delta + 81\alpha\mathbf{v}_1\mathbf{v}_3^{17}\vartheta\delta + \delta^2 + \vartheta\delta^2 + \vartheta^2\delta + \vartheta^2 + \\
& \quad 2\vartheta^2\mathbf{v}_3^9 + \vartheta^2\mathbf{v}_3^{18} + 6561\alpha^2\mathbf{v}_1^2\mathbf{v}_3^{16} + 2\vartheta^2\mathbf{v}_3^9\delta + \delta + 81\alpha\mathbf{v}_1\mathbf{v}_3^8 + \vartheta\mathbf{v}_3^9\delta^2 - \\
& \quad 9\lambda_1\vartheta\mathbf{v}_1\mathbf{v}_3^8\delta = 0)) \wedge \\
& 0 < \vartheta) \wedge 0 < \gamma_0) \wedge 0 < \mu) \wedge 0 < \delta \wedge 0 < \alpha)).
\end{aligned}$$

Fig. 2. The input formulae φ_2 and φ_9 generated by the Maple script in Figure 1. The Maple variables `vv1`, …, `vv3` generated by the function `DEHopfexistence` in the script are typeset as $\mathbf{v}_1, \ldots, \mathbf{v}_3$ here.

Table 1. Applying regular quantifier elimination to the existential closures $\exists\varphi_2$, …, $\exists\varphi_{10}$ of the formulae in Section 3.1. We systematically try several switch settings in REDLOG.

n	`rlsifaco`	`rlqeprecise`	`rlqevarseltry`	result	time (s)
2	○	○	○	SIGXCPU	> 600
3	○	○	○	SIGXCPU	> 600
4	○	○	○	SIGXCPU	> 600
4	○	○	●	false	113.26
4	●	○	●	false	84.10
5	○	○	○	SIGXCPU	> 600
6	○	○	○	SIGXCPU	> 600
6	●	○	●	false	170.10
7	○	○	○	SIGXCPU	> 600
8	○	○	○	SIGXCPU	> 600
8	○	○	●	false	407.63
8	●	○	●	false	346.77
9	○	○	○	SIGXCPU	> 600
10	○	○	○	SIGXCPU	> 600
10	●	○	●	true	543.82

either "true" or "false" depending on whether or not Hopf bifurcations exist. We are going to algorithmically treat the positive cases in more detail later on.

Applying Regular Quantifier Elimination Using the regular quantifier elimination of REDLOG or QEPCAD with their default settings we could not decide any of the existential sentences. Systematically trying some REDLOG switches, we managed to determine special settings that deliver results also for $n = 4, 6, 8, 10$. Table 1 summarizes this experiment, where all rows with unsuccessful non-standard switch settings have been deleted. The keyword SIGXCPU in the result column indicates that the computation has been automatically interrupted after exceeding our chosen limit of 10 minutes of CPU time. The switch `rlsifaco` ("simplifier factorization with ordering relations") toggles the factorization of left hand side polynomials of ordering inequalities for the purpose of simplification; `rlqeprecise` ("quantifier elimination using precise test points") avoids to some extent the substitution of non-standard symbols with virtual substitution; `rlqevarseltry` ("quantifier elimintion variable selection try systematically") does not only apply the standard heuristics for choosing the next variable to be eliminated but additionally breaks tries by trying all best choices and continuing with the smallest result. By default all theses switches are off—for good reason, since this provides in general the by far best performance. Altogether, we consider these results not at all satisfactory: a regular user of some software must not be expected to run lengthy experiments with switch settings in order to hopefully obtain some results.

Table 2. Applying positive quantifier elimination to the existential closures $\exists\varphi_2$, …, $\exists\varphi_{10}$ of the formulae in Section 3.1. We use the standard switch settings in REDLOG; `rlsifaco` is switched on, which is a general recommendation for positive quantifier elimination.

n	$\exists\varphi_n$	$\exists\varphi_n[\lambda_1 \leftarrow -\lambda_1]$	$\exists\varphi_n[\lambda_1 \leftarrow 0]$	time (s)	$\exists\varphi[\lambda_1 \leftarrow \lambda_1 - \lambda_1']$	time (s)
2	false	false	false	< 0.01	SIGXCPU	> 600
3	false	false	false	19.28	SIGXCPU	> 600
4	false	false	false	21.58	SIGXCPU	> 600
5	false	false	false	19.09	SIGXCPU	> 600
6	false	false	false	23.72	SIGXCPU	> 600
7	false	false	false	23.89	SIGXCPU	> 600
8	false	false	false	22.35	SIGXCPU	> 600
9	true	false	false	0.17	true	69.10
10	true	false	false	0.17	true	69.19

Applying Positive Quantifier Elimination Our idea is now to improve the situation by making use of the observation that all variables and parameters except λ_1 are known to be positive. This observation has in fact been used also in [1] by solving for λ_1 in one of the linear equations involving this parameter, so that all remaining variables and parameters are known to be positive and then performing a "hand analysis."

Our approach avoids any hand analysis, because by using the generally applicable case distinction approach on λ_1, cf. Sect. 2.2, we can solve each of the examples in less than half a minute of computation time, cf. the left part of Table 2. Our results for $n = 2, \dots, 9$ confirm those in [1]; for our additional case $n = 10$, we discover the existence of a Hopf bifurcation fixed point. The computation times are the sums of times for all three respective eliminations. For this example, however, the cases $\lambda_1 < 0$ and $\lambda_1 = 0$ turn out to take no considerable time.

The right hand side part of Table 2 summarizes the alternative approach introducing instead of a case distinction a new variable λ_1' and substituting $\lambda_1 \leftarrow \lambda_1 - \lambda_1'$. It performs considerably worse on these examples. This might appear not surprising from the point of view that the complexity of our elimination procedure is exponential in the number of quantified variables. On the other hand, however, iterating case distinctions is exponential as well.

Obtaining Sample Points For the cases in which a Hopf bifurcation fixed point exists the extended quantifier elimination [11] in REDLOG computes in addition a "sample point" fulfilling the existential quantifiers. For the following experiments, we turn on the switch `rlqeaprecise` ("quantifier elimination with answer using precise test points"), which is the analogue for extended quantifier elimination of `rlqeprecise` discussed above. It reduces the introduction of infinitesimals to some extent though not completely. In addition, we switch off `rlsiexpla` ("simplifier explode always") so that atomic formulae are not split by polynomial factorization unless the Boolean structure is preserved.

For the system with $n = 9$ we obtain within 0.11 s the following sample point:

$$\begin{aligned}
\alpha &= 1\\
\gamma_0 &= \frac{-2 \cdot \sqrt{1180986} - 2187 \cdot \varepsilon_1 + 2187}{2187}\\
\delta &= 1\\
\lambda_1 &= L(\varepsilon_1, \varepsilon_2)\\
\mu &= M(\varepsilon_1, \varepsilon_2)\\
\vartheta &= T(\varepsilon_1, \varepsilon_2)\\
\mathbf{v}_1 &= \frac{-2 \cdot \sqrt{1180986} - 2187 \cdot \varepsilon_1 + 2187}{43048908}\\
\mathbf{v}_2 &= 3\\
\mathbf{v}_3 &= 3.
\end{aligned}$$

Here the ε_1, ε_2 are positive infinitesimals, which have to be interpreted as follows: There are $0 < \varepsilon_1, \varepsilon_2 \in \mathbb{R}$, which yield valid sample points, and moreover all positive choices less than ε_1 and ε_2, respectively, yield valid sample points too. The symbols L, M, and T denote rather complicated algebraic expressions, which we present in Appendix A.

In order to get an idea about approximate solutions we set $\varepsilon_1 = \varepsilon_2 = 0$, which yields:

$$\begin{aligned}
\gamma_0 &= 0.00618948546946, & \lambda_1 &= 9540696.11947,\\
\vartheta &= 0.0000571304095036, & \mathbf{v}_1 &= 0.000000314442464411.
\end{aligned}$$

For the case $n = 10$ the same procedure yields within 0.09 s the following approximate sample point, where again γ_0, λ_1, ϑ, and $\mathbf{v}_1$ are approximated by fixing two infinitesimals to 0:

$$\begin{aligned}
\alpha &= 1\\
\gamma_0 &= 0.0100554964908\\
\delta &= 1\\
\lambda_1 &= 17617230.5528\\
\mu &= 0\\
\vartheta &= 0.0000211443608455\\
\mathbf{v}_1 &= 0.000000170287832189\\
\mathbf{v}_2 &= 3\\
\mathbf{v}3 &= 3.
\end{aligned}$$

One can now systematically enumerate further solutions by adding to the input formulae conditions prohibiting for one or several variables or parameters the solutions already found.

3.2 Mass Action Systems

As an example of a mass action system we consider the system investigated in [2, Example 2.1]. It is described by the following system of differential equations:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}x_1(t) &= k_{21}x_1(t)^2x_2(t) + k_{46} - k_{64}x_1(t) - k_{34}x_1(t) + k_{43}\\
\frac{\mathrm{d}}{\mathrm{d}t}x_2(t) &= -k_{21}x_1(t)^2x_2(t) + k_{56} - k_{65}x_2(t)\\
\frac{\mathrm{d}}{\mathrm{d}t}x_3(t) &= k_{34}x_1(t) - k_{43}x_3(t). \qquad (2)
\end{aligned}$$

Again only positive values for the variables are of interest. The automatic generation of a first-order formula stating the question on the existence of Hopf bifurcation fixed points and taking into account the positivity conditions is straightforward. Figure 3 shows our Maple script for this.

The script produces the first-order input formula in Figure 4, which gives the exact, i.e. both necessary and sufficient condition for the existence of a Hopf bifurcation fixed point.

The original literature [2], in contrast, does less: it first derives a necessary condition, and then in a another analysis a sufficient one. To illustrate the difference, notice that "true" is always necessary and "false" is always sufficient but neither of them is generally necessary and sufficient.

Our formulation uses the original coordinates and does not use a coordinate transform as in [2]. Of course, such a coordinate transform could also be realized with our approach in a preprocessing step.

Tables 3 and 4 show the timings and results for regular and for positive quantifier elimination, respectively, on the existential closure of φ. Most surprisingly, on this example regular quantifier elimination performs considerably better than its positive variant. We consider this a rare exception. Anyway it will not be completely ignored: We expect from a careful analysis of the various quantifier elimination runs some insights and ideas for improvements of the procedures. There is another annoying fact, which cannot be ignored: In the previous section we had found that switching on `rlsifaco` ("simplifier factorization with ordering relations") and switching off `rlqeprecise` ("quantifier elimination using precise test points") and `rlqevarseltry` ("quantifier elimination variable selection try systematically") should be the default choice for positive quantifier elimination. The 5th row in Table 4 shows that this very choice fails on this particular example. Notice, however, that things look much better when experimentally switching on `rlqevarseltry`. This indicates that the unusual behavior is mainly caused by the fact that good variables orderings cannot be easily recognized.

Anyway, good news is that we can decide the existence of a Hopf bifurcation fixed point within a few seconds. As the answer is affirmative, also sample answer points can be computed. For this we use regular extended quantifier elimination with the most efficient switch settings given in the 4th row of Table 3. We obtain

```
eq11 := diff(x1(t),t)=k21*x1(t)^2*x2(t)+k46-k64*x1(t)-k34*x1(t)+k43*x3(t);
eq12 := diff(x2(t),t)=-k21*x1(t)^2*x2(t)+k56-k65*x2(t);
eq13 := diff(x3(t),t)=k34*x1(t)-k43*x3(t);
fcns1 := {x1(t),x2(t),x3(t)};
params1 := [k21,k46,k64,k34,k43,k56,k65];
paramcondlist1 := [k21>0, k46>0, k64>0, k34>0, k43>0, k56>0, k65>0];
funccondlist1 := [x1(t)>0, x2(t)>0, x3(t)>0];
DEHopfexistence({eq11,eq12,eq13},
   fcns1,params1,funccondlist1,paramcondlist1);
```

Fig. 3. A Maple script for generating the first-order input formula φ for Section 3.2.

$$\begin{aligned}
\varphi \equiv\ & \exists \mathbf{v}_3 \exists \mathbf{v}_2 \exists \mathbf{v}_1 (((0 < \mathbf{v}_1 \wedge 0 < \mathbf{v}_2) \wedge 0 < \mathbf{v}_3) \wedge \\
& ((((((((((k_{21}\mathbf{v}_1^2\mathbf{v}_2 + k_{46} - k_{64}\mathbf{v}_1 - k_{34}\mathbf{v}_1 + k_{43}\mathbf{v}_3 = 0 \wedge \\
& -k_{21}\mathbf{v}_1^2\mathbf{v}_2 + k_{56} - k_{65}\mathbf{v}_2 = 0) \wedge \\
& k_{34}\mathbf{v}_1 - k_{43}\mathbf{v}_3 = 0) \wedge \\
& (0 < k_{64}k_{65}k_{43} + k_{64}k_{21}\mathbf{v}_1^2k_{43} - 2k_{21}\mathbf{v}_1\mathbf{v}_2k_{65}k_{43} \wedge \\
& k_{21}\mathbf{v}_1^2k_{43}^2 + k_{21}^2\mathbf{v}_1^4k_{43} + k_{21}^2\mathbf{v}_1^4k_{34} + 2k_{34}k_{65}k_{43} + 2k_{64}k_{34}k_{65} + 2k_{64}k_{65}k_{43} + \\
& \quad k_{64}^2k_{21}\mathbf{v}_1^2 + k_{21}^2\mathbf{v}_1^4k_{64} + k_{34}^2k_{21}\mathbf{v}_1^2 + k_{34}k_{64}k_{43} + k_{64}k_{43}^2 + k_{65}k_{43}^2 + k_{64}k_{65}^2 + \\
& \quad k_{65}^2k_{43} + k_{34}k_{65}^2 + k_{64}^2k_{65} + k_{64}^2k_{43} + 2k_{64}k_{21}\mathbf{v}_1^2k_{43} + 2k_{34}k_{21}\mathbf{v}_1^2k_{43} + \\
& \quad 2k_{21}\mathbf{v}_1^2k_{64}k_{65} + 2k_{21}\mathbf{v}_1^2k_{65}k_{43} + 2k_{21}\mathbf{v}_1^2k_{34}k_{65} + 2k_{64}k_{34}k_{21}\mathbf{v}_1^2 - \\
& \quad 4k_{21}^2\mathbf{v}_1^3\mathbf{v}_2k_{43} - 2k_{21}^2\mathbf{v}_1^3\mathbf{v}_2k_{64} - 2k_{21}^2\mathbf{v}_1^3\mathbf{v}_2k_{34} + 4k_{21}^2\mathbf{v}_1^2\mathbf{v}_2^2k_{43} + 4k_{21}^2\mathbf{v}_1^2\mathbf{v}_2^2k_{65} - \\
& \quad 2k_{21}^2\mathbf{v}_1^3\mathbf{v}_2k_{65} - 2k_{21}\mathbf{v}_1\mathbf{v}_2k_{43}^2 - 2k_{21}\mathbf{v}_1\mathbf{v}_2k_{65}^2 + k_{34}^2k_{65} - 4k_{21}\mathbf{v}_1\mathbf{v}_2k_{65}k_{43} - \\
& \quad 4k_{21}\mathbf{v}_1\mathbf{v}_2k_{64}k_{65} - 4k_{21}\mathbf{v}_1\mathbf{v}_2k_{64}k_{43} - 4k_{21}\mathbf{v}_1\mathbf{v}_2k_{34}k_{65} - 2k_{34}k_{21}\mathbf{v}_1\mathbf{v}_2k_{43} = 0)) \wedge \\
& 0 < k_{21}) \wedge 0 < k_{46}) \wedge 0 < k_{64}) \wedge 0 < k_{34}) \wedge 0 < k_{43}) \wedge 0 < k_{56}) \wedge 0 < k_{65}))
\end{aligned}$$

Fig. 4. The formula φ generated by the Maple script in Figure 3. The Maple variables `vv1`, …, `vv3` generated by the function `DEHopfexistence` in the script are typeset as $\mathbf{v}_1, \ldots, \mathbf{v}_3$ here.

Table 3. Applying regular quantifier elimination to the existential closures $\exists\varphi$ of the formula in Section 3.2. We systematically try several switch settings in REDLOG.

`rlsifaco`	`rlqeprecise`	`rlqevarseltry`	result	time (s)
○	○	○	SIGXCPU	> 600
○	○	●	true	261.72
○	●	○	true	3.03
○	●	●	true	1.54
●	○	○	SIGXCPU	> 600
●	○	●	true	264.73
●	●	○	true	3.35
●	●	●	true	1.79

Table 4. Applying positive quantifier elimination to the existential closures $\exists\varphi$ of the formula in Section 3.2. We systematically try several switch settings in REDLOG.

`rlsifaco`	`rlqeprecise`	`rlqevarseltry`	result	time (s)
○	○	○	true	20.79
○	○	●	true	32.20
○	●	○	SIGXCPU	> 600
○	●	●	SIGXCPU	> 600
●	○	○	SIGXCPU	> 600
●	○	●	true	32.54
●	●	○	SIGXCPU	> 600
●	●	●	true	8.03

after 3.05 s:

$$\begin{array}{lllll} k_{21} = 32, & k_{34} = 2/3, & k_{43} = 1, & k_{46} = 1/12, & k_{56} = 1, \\ k_{64} = 14/3, & k_{65} = 1/2, & \mathbf{v}_1 = 1/8, & \mathbf{v}_2 = 1, & \mathbf{v}_3 = 1/12. \end{array}$$

So for this example, the switch `rlqeaprecise` is powerful enough to entirely avoid the introduction of infinitesimals. Again, further sample points can be enumerated by explicitly excluding in the input formula the ones already found.

4 Conclusions and Future Work

Using generic methods to reduce the Hopf bifurcation problem to a quantifier elimination problem and then using quantifier elimination methods available in REDLOG we can reconstruct most of the results given in the literature within less than a minute of computation time.

We have restricted ourselves to examples already treated in the literature by other symbolic methods. However, we presume that our generic methods are applicable not only to many other problems in theory but also in practice. As no tedious hand computations are involved the use of these generic methods will hopefully be a useful tool for investigating a wide variety of other examples. In future work we will test our generic methods for parametrically investigating Hopf bifurcation fixed points on examples treated in the literature by numerical computations or pure hand calculations, e.g. on the systems described in [4–6, 19]. As there is the option to have some parameters fixed and only to investigate few others symbolically—still extending the possibilities of pure numeric investigations—one can also apply these techniques to larger systems, which cannot be expected to be amenable for a full parametric analysis, e.g. the systems described in [7, 8].

On the quantifier elimination side, the great challenge is to improve the procedure such that the original elimination problems, in contrast to the existential closures, become tractable. There is actually good hope for success: In principle the elimination for the closure is a harder problem than that for the original formula, but we apparently benefit from the greater freedom in choosing the variable order for elimination. The fact that this is successful indicates that the problems are not inherently hard.

5 Acknowledgement

We are grateful to A. Dolzmann, C. Groß, A. Reitelmann, and U. Waldmann for helpful discussions.

A Exact Solutions with Infinitesimals for Section 3.1

$$
\begin{aligned}
L(\varepsilon_1,\varepsilon_2) = \Big(&-200156684335646976 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_2 \\
&-22877954716428 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \\
&+5083989936984 \cdot \sqrt{1180986} + 5559342996092004 \cdot \varepsilon_1 \\
&+236405060478973427779392 \cdot \varepsilon_2^2 + 5404230477062468352 \cdot \varepsilon_2 \\
&-5216187623191776\Big)/ \\
&\Big(1062882 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{393662} \\
&+387420489 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_1 \\
&-387420489 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \\
&-172186884 \cdot \sqrt{1180986} \cdot \varepsilon_1 - 83691328896 \cdot \sqrt{1180986} \cdot \varepsilon_2 \\
&+166872690 \cdot \sqrt{1180986} - 94143178827 \cdot \varepsilon_1^2 \\
&-91516468147776 \cdot \varepsilon_1 \cdot \varepsilon_2 + 182475286515 \cdot \varepsilon_1 \\
&+91516468147776 \cdot \varepsilon_2 - 181313497440\Big)
\end{aligned}
$$

$$
\begin{aligned}
M(\varepsilon_1,\varepsilon_2) = \Big(&61010978765184 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{393662} \cdot \varepsilon_2 \\
&+ 22238501759909568 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_1 \cdot \varepsilon_2 \\
&-22238501759909568 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_2 \\
&-2402002240184651776 \cdot \sqrt{1180986} \cdot \varepsilon_2^2 \\
&-2626589449641916717056 \cdot \varepsilon_1 \cdot \varepsilon_2^2 \\
&+2626589449641916717056 \cdot \varepsilon_2^2\Big)/ \\
&\Big(4649045868 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{393662} \cdot \varepsilon_1 \\
&-4649045868 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{393662} \\
&+847288609443 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_1^2 \\
&-1694577218886 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \cdot \varepsilon_1 \\
&+1684121117211 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \\
&-564859072962 \cdot \sqrt{1180986} \cdot \varepsilon_1^2 \\
&-366065872591104 \cdot \sqrt{1180986} \cdot \varepsilon_1 \cdot \varepsilon_2 \\
&+1106473861368 \cdot \sqrt{1180986} \cdot \varepsilon_1 + 366065872591104 \cdot \sqrt{1180986} \cdot \varepsilon_2 \\
&-727577567910 \cdot \sqrt{1180986} - 205891132094649 \cdot \varepsilon_1^3 \\
&-200146515839186112 \cdot \varepsilon_1^2 \cdot \varepsilon_2 + 604964583702954 \cdot \varepsilon_1^2 \\
&+400293031678372224 \cdot \varepsilon_1 \cdot \varepsilon_2 - 1202306669284833 \cdot \varepsilon_1 \\
&-397823091334329024 \cdot \varepsilon_2 + 790681240245960\Big)
\end{aligned}
$$

$$
\begin{aligned}
T(\varepsilon_1,\varepsilon_2) = \Big(&-6561 \cdot \sqrt{\varepsilon_1} \cdot \sqrt{4 \cdot \sqrt{1180986} + 2187 \cdot \varepsilon_1} \cdot \sqrt{3} \\
&+1458 \cdot \sqrt{1180986} + 1594323 \cdot \varepsilon_1 \\
&+1549839424 \cdot \varepsilon_2 - 1495912\Big)/1549839424
\end{aligned}
$$

References

1. Boulier, F., Lefranc, M., Lemaire, F., Morant, P., Ürgüplü, A.: On proving the absence of oscillations in models of genetic circuits. In Anai, H., Horimoto, H., Kutsia, T., eds.: Algebraic Biology (AB 2007). Volume 4545 of Lecture Notes in Computer Science. Springer-Verlag (2007) 66–80
2. Gatermann, K., Eiswirth, M., Sensse, A.: Toric ideals and graph theory to analyze Hopf bifurcations in mass action systems. Journal of Symbolic Computation **40**(6) (2005) 1361–1382

3. Niu, W., Wang, D.: Algebraic approaches to stability analysis of biological systems. Mathematics in Computer Science **1**(3) (2008) 507–539
4. Sensse, A., Hauser, M.J.B., Eiswirth, M.: Feedback loops for Shilnikov chaos: The peroxidase-oxidase reaction. The Journal of Chemical Physics **125** (2006) 014901–1–12
5. Fussmann, G.F., Ellner, S.P., Shertzer, K.W., Hairston, Nelson G., J.: Crossing the Hopf bifurcation in a live predator-prey system. Science **290**(5495) (2000) 1358–1360
6. Mincheva, M., Roussel, M.R.: Graph-theoretic methods for the analysis of chemical and biochemical networks. I. multistability and oscillations in ordinary differential equation models. Journal of Mathematical Biology **55**(1) (2007) 61–86
7. Novak, B., Pataki, Z., Ciliberto, A., Tyson, J.J.: Mathematical model of the cell division cycle of fission yeast. Chaos: An Interdisciplinary Journal of Nonlinear Science **11**(1) (2001) 277–286
8. Tyson, J.J., Chen, K., Novak, B.: Network Dynamics and Cell Physiology. Nat Rev Mol Cell Biol **2**(12) (2001) 908–916
9. El Kahoui, M., Weber, A.: Deciding Hopf bifurcations by quantifier elimination in a software-component architecture. Journal of Symbolic Computation **30**(2) (August 2000) 161–179
10. Weber, A.: Quantifier elimination on real closed fields and differential equations. In Löwe, B., ed.: Algebra, Logic, Set Theory – Festschrift für Ulrich Felgner zum 65. Geburtstag. Volume 4 of Studies in Logic. College Publications (2007) 291–315
11. Weispfenning, V.: Simulation and optimization by quantifier elimination. Journal of Symbolic Computation **24**(2) (August 1997) 189–208 Special issue on applications of quantifier elimination.
12. Dolzmann, A., Sturm, T.: Redlog: Computer algebra meets computer logic. ACM SIGSAM Bulletin **31**(2) (June 1997) 2–9
13. Weispfenning, V.: Quantifier elimination for real algebra—the quadratic case and beyond. Applicable Algebra in Engineering Communication and Computing **8**(2) (February 1997) 85–101
14. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. Journal of Symbolic Computation **12**(3) (September 1991) 299–328
15. Sturm, T.: Redlog online resources for applied quantifier elimination. Acta Academiae Aboensis, Ser. B **67**(2) (2007) 177–191
16. Dolzmann, A., Sturm, T.: Simplification of quantifier-free formulae over ordered fields. Journal of Symbolic Computation **24**(2) (August 1997) 209–231
17. Dolzmann, A., Sturm, T., Weispfenning, V.: A new approach for automatic theorem proving in real geometry. Journal of Automated Reasoning **21**(3) (1998) 357–380
18. Brown, C.W., Groß, C.: Efficient preprocessing methods for quantifier elimination. In Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V., eds.: Computer Algebra in Scientific Computing (CASC '06). Volume 4194 of Lecture Notes in Computer Science., Chisinau, Moldova, Springer-Verlag (September 2006) 89–100
19. Sun, M., Tian, L., Yin, J.: Hopf bifurcation analysis of the energy resource chaotic system. International Journal of Nonlinear Science **1**(1) (2006) 49–53