An algorithmic version of Zariski's lemma*

Franziskus Wiesnet[1,2,3]

[1] Ludwig-Maximilians Universität, Theresienstr. 39, 80333 München, Germany
wiesnet@mathematik.uni-muenchen.de
[2] Università degli Studi di Trento, Via Sommarive 14 - 38123 Povo, Italy
franziskus.wiesnet@unitn.it
[3] Università degli studi di Verona, Strada le Grazie 15 - 37134 Verona, Italy

Abstract. Zariski's lemma was formulated and used by Oscar Zariski to prove Hilbert's Nullstellensatz. This article gives an elementary and constructive proof of Zariski's lemma and only uses basics of integral ring extensions under the condition that each field is discrete. After this constructive proof we take a look at the computational side. We give a computational interpretation of Zariski's lemma and use our constructive proof to develop an algorithm which realises the computational interpretation. This is a typical approach in constructive mathematics.

Keywords: Zariski's lemma, constructive algebra, computational algebra, program extraction, proof mining

1 Introduction

1.1 Historical background

Presumably the first time Zariski's lemma appeared was in [19]. There Oscar Zariski used it to prove Hilbert's Nullstellensatz. In 1976, John McCabe gave an interesting but not constructive proof [9], which relied on the existence of maximal ideals. In 2020 Daniel Wessel has avoided this maximality argument by using Jacobson radicals [16]. However, the proof still contains a non-constructive moment. To wit, if R is an algebra over a field K and $S \subseteq R$ is a finite subset, then there exists $S_0 \subseteq S$ maximal such that all elements in S_0 are algebraically independent over K. To avoid this, one could use Noether normalization. A constructive proof of Noether normalisation is given in [10] and Zariski's lemma is a corollary of it [5, 10, 13]. The proofs in [1, 2, 15, 19] are non-constructive but, instead of a maximal algebraically independent subset, they use induction on the number of generators of the algebra. This will also be part of our constructive proof. The proof in the present paper is a direct and constructive proof of Zariski's lemma. To get this proof, we have analysed the proofs in the sources above and put them together with some new ideas.

* I would like to thank the Istituto Nazionale di Alta Matematica "Francesco Severi" for the financial support of my PhD study. Thanks for direct support goes to Daniel Wessel for his ideas and taking a look at the manuscript, my supervisor Peter Schuster for the selection of this topic and support of the publication, and Henri Lombardi and Ihsen Yengui who helped to improve the proof with important comments.

1.2 Method of proof interpretation

We have considered some non-constructive proofs of Zariski's lemma, analysed them and rebuilt them into a new constructive proof (Section 2). This approach was inspired by the methods of *proof mining* [6, 7]. Inspired by the methods of the formal *program extraction* from proofs as in [3, 14, 17], we have turned our constructive proof into algorithms and realisability statements (Section 3). But in contrast to formal program extraction, when we speak about "realisability" we do not mean the rigorously defined realisability predicate of program extraction, for example given in [14]. In this paper "realisability" is rather a heuristic notion.

Our approach shows a typical approach in constructive mathematics. Analysing a theorem constructively often goes as follows:

- Formulate a *quite* constructive proof of the theorem.
- Formulate an algorithmic interpretation of the theorem.
- Inspired by the quite constructive proof formulate an algorithm which shall realise the algorithmic interpretation.
- Prove that the algorithm is indeed a realiser of the algorithmic interpretation.

This paper is an example where these steps are done manually on paper and where the formulation of the quite constructive proof is only necessary to get an inspiration for the other steps. As the space in this paper is quite scarce we have to forgo the fourth step. In particular, we do not give proofs in Section 3. However, in the example of program extraction from proofs above usually only the quite constructive proof is formulated manually and the other steps are done by the computer. Note that we have written "quite constructive" because sometimes one can bypass a non-constructive moment or it can be included as assumption in the algorithmic version. We also see an example of this in the present paper: since our proof uses case distinction on $x = 0$ or $x \neq 0$ for all x in a ring, we assume that this ring is discrete. However, this is the only computational restriction we have to make.

1.3 Fundamental notions

Before formulating a proof of Zariski's lemma and the computational interpretation, we define the underlying objects. In Zariski's lemma, we use axioms for rings, field and algebra and their structures. But an algorithm cannot operate on axioms. More specifically: if we state an algorithm about a field, we do not use the field axioms in the algorithm but we use the field structure like $+$, $\cdot$, 0, 1 and so on. Therefore, we first define the underlying structures precisely:

In our setting a *ring structure* $(R, +, \cdot, 0, 1, -, =)$ is a set R equipped with an addition operator $+ : R \times R \to R$, a multiplication operator $\cdot : R \times R \to R$, a zero element $0 \in R$, an unit element $1 \in R$, an additive inverse function $- : R \to R$ and an equality $= \subseteq R \times R$. If furthermore $=$ is an equivalence relation and compatible with $+, \cdot, -$, i.e. $=$ is a congruence relation on $(R, +, \cdot, 0, 1, -)$, and the other ring axioms are fulfilled (w.r.t. the equality $=$), R is a *ring*. In our case a ring is always commutative. We call $(K, +, \cdot, 0, 1, -, ^{-1}, =)$ a *field structure* if

$(K, +, \cdot, 0, 1, -, =)$ is a ring structure and $^{-1} : K \to K$ is a map. If K is a ring, $xx^{-1} = 1 \vee x = 0$ for all $x \in K$ and $1 \neq 0$, K is a *field*.

Since the notation of $+, \cdot, 0, 1, -, ^{-1}$ and $=$ will not change, we do not mention it and say that R is a ring (structure) or K is a field (structure) and so on. A *homomorphism* $\phi : R \to S$ between two ring structures R and S is a map which preserves the structure in the canonical way.

For a ring structure R we define the *ring structure of polynomials* $R[X]$ with coefficients in R by the well-known construction. For $n \in \mathbb{N}$ we have also the polynomial ring structure in n variables denoted by $R[X_1, \ldots, X_n]$. Obviously, if R is a ring then so is $R[X_1, \ldots, X_n]$.

An *algebra structure* R over a field structure K, or short K-algebra structure, is a ring structure together with a map $K \to R$. If R is a ring, K is a field and the map $K \to R$ is a homomorphism, we call R a *K-algebra*. For a K-algebra R and $x_1, \ldots, x_n \in R$ we get an extension $K[X_1, \ldots, X_n] \to R$ of the homomorphism by $X_i \mapsto x_i$. We denote the image by $K[x_1, \ldots, x_n]$, where an element is in the image of a homomorphism if it is equal (w.r.t. $=$) to a value of the homomorphism.

The following definition comes from [8, 18]:

Definition 1. *A ring structure R is* discrete *if all its operators are computable. Here $=$ is seen as a Boolean-valued function. A field structure K is* discrete *if it is discrete as ring and $^{-1}$ is computable.*

Here, "computability" means that we can use the operations above freely in our algorithms. In particular, we can use the ring operators arbitrarily, and can distinguish between the cases $x = y$ and $x \neq y$.

We do not specify the underlying theory of computability and how the objects are represented, as there are several possibilities. However, in Section 3 we develop an algorithm out of the constructive proof. If one wants this algorithm to be a Turing machine, a discrete structure should be interpreted as a structure where all operators (including $=$) are representable by a Turing machine.

In this article we tacitly assume that each structure be discrete and make case distinctions like $x = 0 \vee x \neq 0$ without explicitly justifying them.

Remark 1. If K is a discrete field structure then the polynomial ring structure $K[X_1, \ldots, X_n]$ is also discrete and for $f \in K[X_1, \ldots, X_n]$ we can decide whether $f \in K$ or $f \notin K$ because $f \in K$ if and only if all non-constant coefficients are zero. Similarly, it is even possible to compute $\deg(f)$ for every $f \in K[X]$.

Let $A \subseteq B$ be a ring extension, i.e. the inclusion $A \to B$ is a homomorphism. An element $x \in B$ is called *integral* over A if there are $a_0, \ldots, a_{k-1} \in A$ such that $x^k + a_{k-1}x^{k-1} + \cdots + a_0 = 0$. The ring extension $A \subseteq B$ is called *integral*, if each $x \in B$ is integral over A.

In our constructive proof we need the following two lemmas. The proofs of them we refer to are also constructive.

Lemma 1. *If $A \subseteq B$ is an integral ring extension and B is a field then A is a field, too.*

Proof. A constructive proof is given in [1, Proposition 5.7]. □

Lemma 2. *Let $A \subseteq B$ a ring extension. If $x_1, \dots, x_n \in B$ are integral over A then the ring extension $A \subseteq A[x_1, \dots, x_n]$ is integral.*

Proof. This follows from Corollary 5.3 of [1]. □

2 A constructive proof

In this section we give a new constructive proof of Zariski's lemma. The proof does not use any non-constructive principles (except that the rings be discrete). In the next section we use this proof as basis to create an algorithmic version.

Theorem 1 (Zariski's lemma). *Let K be a field and R an algebra over K which is a field. Suppose that $R = K[x_1, \dots, x_n]$ for some $x_1, \dots, x_n \in R$. Then $x_1, \dots x_n$ are algebraic over K, i.e. there are $f_1, \dots, f_n \in K[X] \setminus K$ with $f_i(x_i) = 0$ for all i.*

Proof. If $n = 0$, there is nothing to show. We continue by considering the case $n = 1$: if $x_1 = 0$ then $R = K$ and we are done. Otherwise, x_1 is invertible. Since R is a field, there is $p \in K[X] \setminus \{0\}$ with $x_1 p(x_1) = 1$. We set $q := Xp - 1 \in K[X]$. Then $q \neq 0$ because $\deg(Xp) > 0$ and $q(x_1) = 0$.

Next, we consider the case $n = 2$: We show that x_1 is algebraic. The argument for x_2 is analogous. If $x_2 = 0$ we are done by the case $n = 1$ as above. Otherwise, we have $p \in K[X_1, X_2]$ with $p(x_1, x_2)x_2 = 1$. Therefore, $q := Xp(x_1, X) - 1$ is a polynomial in $K[x_1][X]$ with $q(x_2) = 0$ and $q \neq 0$ as its constant coefficient is -1. Let $y \in K[x_1]$ be the leading coefficient of q, which is non-zero by definition. Then $K[x_1, y^{-1}] \subseteq K[x_1, x_2]$ is an integral ring extension by Lemma 2 because x_2 is integral over $K[x_1, y^{-1}]$ witnessed by $y^{-1}q \in K[x_1, y^{-1}][X]$. Therefore, $K[x_1, y^{-1}]$ is a field by Lemma 1.

With this preparation we are now able to construct a non-zero polynomial with root x_1. By $y \in K[x_1]$, there is $f \in K[X]$ such that $f(x_1) = y$. If $f \in K$ then $K[x_1, y^{-1}] = K[x_1]$ and we are done by the case $n = 1$. So, we assume $f \in K[X] \setminus K$. If $1 - f(x_1) = 0$ then x_1 is algebraic over K. Otherwise, $1 - f(x_1)$ is invertible[4] in $K[x_1, y^{-1}]$ and therefore there is $h \in K[X]$ and $N \in \mathbb{N}$ with $(1 - f(x_1))^{-1} = h(x_1)y^{-N} = h(x_1)f(x_1)^{-N}$. So, we have

$$f(x_1)^N - h(x_1)(1 - f(x_1)) = 0.$$

It remains to show that $f^N - h(1 - f) \neq 0$ in $K[X]$. By the binomial theorem there is a $g \in K[X]$ with $f^N = 1 + (1 - f)g$, and so

$$f^N - h(1 - f) = 1 + (1 - f)(g - h).$$

[4] The idea to take $1 - f(x_1)$ is based on an idea by Daniel Wessel [16] and an hint by Henri Lombardi. Inspired by [15], the first approach of the author was to take $g(x_1)$ for some irreducible $g \in K[X]$ with $g \nmid f$.

Since f is non-constant, also $1-f$ is non-constant. Now assume that $f^N - h(1-f) = 0$ then $g-h = 0$ as otherwise $\deg((1-f)(g-h)) > 0$ and $1+(1-f)(g-h) \neq 0$. But then $0 = 1$, a contradiction.

Finally, we assume $n \geq 2$ and use induction over n. The base case $n = 2$ was done above. For the induction step let $n \geq 3$ be given. Again, we just show that x_1 is algebraic. The arguments for $x_2, \ldots, x_n$ are analogous. Let $L := K(x_1)$ the field of fractions of $K[x_1]$. Since R is a field, we can consider $L \subseteq R$ and therefore $L[x_2, \ldots, x_n] = R$. By induction, each x_i for $i \in \{2, \ldots, n\}$ is algebraic over L. So for each such i, there is a monic polynomial $f_i \in L[X]$ with $f_i(x_i) = 0$. Let v_i be the product of the denominators of all coefficients in f_i and $v := \prod_{i=2}^{n} v_i$. Then all x_i are integral over $K[x_1, v^{-1}]$. Using Lemma 2, $K[x_1, v^{-1}] \subseteq K[x_1, \ldots, x_n]$ is an integral ring extension. By Lemma 1, also $K[x_1, v^{-1}]$ is a field. By the case $n = 2$ it follows that x_1 is algebraic over K. □

3 Computational interpretation

The goal of this section is to build an algorithm out of the constructive proof above. One could argue that this is not necessary as a constructive proof provides an algorithm by definition and it is an easy exercise to extract it. However, as we are not using computer support and the proof is not totally formal, there is still some work to do. In particular, we consider the concepts we have used in the proof and give them a computational meaning in the next two definitions.

3.1 Preliminary

We use the following syntactical abbreviations: $\vec{x} := x_1, \ldots, x_n$; $\vec{X} := X_1, \ldots, X_n$; $\vec{y} := y_1, \ldots, y_m$ and $\vec{Y} := Y_1, \ldots, Y_m$. For $n \in \mathbb{N}$ and any $I \in \mathbb{N}^n$ we define $\vec{x}^I := \prod_{i=1}^{n} x_i^{I_i}$ and $\vec{X}^I := \prod_{i=1}^{n} X_i^{I_i}$.

In Zariski's lemma a K-algebra $K[\vec{x}]$ is given. In particular, there is a surjective homomorphism from $K[\vec{X}]$ to $K[\vec{x}]$. It is well-known that the existence of a right-inverse of a surjection in general requires the axiom of choice. That is the reason why we do not use it computationally and we work on the level of the polynomial rings. The following definition is the computational interpretation of $K[\vec{y}] \subseteq K[\vec{x}]$ being a ring extension on the level of polynomials:

Definition 2. *Let K be a field, R be a K-algebra and $\vec{x}, \vec{y} \in R$. We say that $K[\vec{y}] \subseteq K[\vec{x}]$ is a* ring extension of K-algebras *witnessed by $\vec{h} := h_1, \ldots, h_m \in K[\vec{X}]$ if $h_i(\vec{x}) = y_i$ for all i. In short notation we write $\vec{h}(\vec{x}) = \vec{y}$.*

Similarly, the next definition is the computational interpretation of $K[\vec{x}]$ being a field on the level of polynomials:

Definition 3. *Let a field K, a K-algebra R and $\vec{x} \in R$ be given. A computable function $\iota : K[\vec{X}] \to K[\vec{X}]$ with $f(\vec{x}) = 0 \vee (\iota(f))(\vec{x}) f(\vec{x}) = 1$ for all $f \in K[\vec{X}]$ is called* algebraic inverse function *on $K[\vec{x}]$.*

Remark 2. An algebraic inverse function does not have to be compatible with the equality relation of the ring structure $K[\vec{X}]$. From an algebraic inverse function on $K[\vec{x}]$ and a right inverse of a surjection $K[\vec{X}] \to K[\vec{x}]$ we get that $K[\vec{x}]$ is a field. But this is constructively delicate, so in both definitions above we have avoided a direct use of $K[\vec{x}]$ and we also do this in the following algorithms. The occurrence of $K[\vec{x}]$ in the definitions above is just a way of speaking.

Similar to above, "computable" means that we can use the algebraic inverse function freely in our algorithm. For instance, if the algorithm shall be a Turing machine, an algebraic inverse function has to be Turing computable.

In the light of the definitions above: an algorithm which realises Zariski's lemma takes an algebraic inverse function on $K[\vec{x}]$ as input and returns polynomials $f_1, \dots, f_n \in K[X] \setminus K$ with $f_i(x_i) = 0$ for all $i \in \{1, \dots, n\}$.

3.2 Some algorithms for integral extensions of algebras

The following lemma is an algorithmic version, in terms of algebras over a field, of Lemma 2. Given a field K, R be a K-algebra and $\vec{x}, \vec{y} \in R$. As realiser of this lemma we expect an algorithm which takes for each x_i an integral equation in the form $P_i(\vec{y})(x_i) = 0$ for some monic $P_i \in K[\vec{Y}][X]$ and some $f \in K[\vec{X}]$ as input and returns an integral equation of $f(\vec{x})$ as output in the form $Q(\vec{y})(f(\vec{x})) = 0$ for some monic $Q \in K[\vec{Y}][X]$.

Algorithm 1. *Given a field structure K, $f \in K[\vec{X}]$ and $k_i \in \mathbb{N}$, $g^{(i)}_{k_i-1}, \dots, g^{(i)}_0 \in K[\vec{Y}]$ for each $i \in \{1, \dots, n\}$. We compute $k \in \mathbb{N}$ and $g_{k-1}, \dots, g_0 \in K[\vec{Y}]$:*

1. *Define $\mathcal{I} := \{I \in \mathbb{N}^n | I_1 < k_i, \dots, I_n < k_n\}$ and for each $I \in \mathcal{I}$ compute the finite sum $f\vec{X}^I = \sum_{J \in \mathbb{N}^n} f_{IJ} \vec{X}^J$ with $f_{IJ} \in K$.*
2. *For each $I \in \mathcal{I}$ and $i \in \{1, \dots, n\}$ replace each $X_i^{k_i}$ by $-g^{(i)}_{k_i-1} X^{k_i-1} - \dots - g^{(i)}_0$ in $\sum_{J \in \mathbb{N}^n} f_{IJ} \vec{X}^J$ one by one until we get a polynomial of the form $\sum_{J \in \mathcal{I}} g_{IJ} \vec{X}^J$ with $g_{IJ} \in K[\vec{Y}]$*
3. *Compute the characteristic polynomial $P \in K[\vec{Y}][X]$ of the matrix $(g_{IJ})_{I,J \in \mathcal{I}}$ as the determinant of the matrix $(\delta_{IJ} X - g_{IJ})_{I,J \in \mathcal{I}}$, where $\delta_{IJ} X := X$ if $I = J$, and $\delta_{IJ} X := 0$ if $I \neq J$.*
4. *Let $P = \sum_{i=0}^{l} g_i X^i$ for some $l \in \mathbb{N}$ and $g_i \in K[\vec{Y}]$. Return $k := \prod_{i=1}^{n} k_i$ and the first k coefficients $g_{k-1}, \dots, g_0$ of P, where $g_i := 0$ if $i > l$.*

Note that in Step 2 there is no order mention in which each X_i has to be replaced. However, the following lemma is true for any possible order.

Lemma 3. *In the situation of Algorithm 1 we assume that K is a field, R is a K-algebra and $\vec{x}, \vec{y} \in R$ with*

$$x_i^{k_i} + g^{(i)}_{k_i-1}(\vec{y}) x_i^{k_i-1} + \dots + g^{(i)}_0(\vec{y}) = 0 \tag{1}$$

for each $i \in \{1, \dots, n\}$. Then

$$(f(\vec{x}))^k + g_{k-1}(\vec{y})(f(\vec{x}))^{k-1} + \dots + g_0(\vec{y}) = 0.$$

The next lemma is an algorithmic version of Lemma 1. In terms of K-algebras and in the light of computational algebra, we want to compute an algebraic inverse function on $K[\vec{y}]$ from an algebraic inverse function on $K[\vec{x}]$ and the integral equations of $\vec{x}$.

Algorithm 2. *Let a field structure K, $\vec{h} := h_1, \ldots, h_m \in K[\vec{X}]$, $\iota : K[\vec{X}] \to K[\vec{X}]$ and $k_i \in \mathbb{N}$, $g^{(i)}_{k_i-1}, \ldots, g^{(i)}_0 \in K[\vec{Y}]$ for each $i \in \{1, \ldots, n\}$ be given. We define a map $\tilde{\iota} : K[\vec{Y}] \to K[\vec{Y}]$ as follows:*

1. *Given an input $f \in K[\vec{Y}]$, compute $p := \iota(f(\vec{h})) \in K[\vec{X}]$.*
2. *Apply Algorithm 1 to K, p and k_i, $g^{(i)}_{k_i-1}, \ldots, g^{(i)}_0$ for each $i \in \{1, \ldots, n\}$ to get $k \in \mathbb{N}$ and $g_{k-1}, \ldots, g_0 \in K[\vec{Y}]$.*
3. *Return $-g_{k-1} - g_{k-2}f - \cdots - g_0 f^{k-1}$.*

Lemma 4. *In the situation of Algorithm 2 we assume that K is a field and let a K-algebra R and $\vec{x}, \vec{y} \in R$ be given such that $K[\vec{y}] \subseteq K[\vec{x}]$ is an extension of K-algebras witnessed by $\vec{h}$. Furthermore, we assume that ι is an algebraic inverse function and*

$$x_i^{k_i} + g^{(i)}_{k_i-1}(\vec{y})x_i^{k_i-1} + \cdots + g^{(i)}_0(\vec{y}) = 0$$

for all $i \in \{1, \ldots, n\}$. Then $\tilde{\iota}$ is an algebraic inverse function on $K[\vec{y}]$.

3.3 An algorithm for Zariski's lemma

In the following we give an algorithmic version of Zariski's lemma. As in the proof of Theorem 1, we first consider the cases $n = 1$ and $n = 2$. Hence, the next two algorithms construct the polynomials which witness that the generators are algebraic.

Algorithm 3. *Given a discrete field structure K, a discrete K-algebra structure R, $x \in R$ and $\iota : K[X] \to K[X]$, we compute an element $f \in K[X]$ as follows:*

1. *If $x = 0$, return X.*
2. *If $x \neq 0$, return $X\iota(X) - 1$.*

Lemma 5. *In the situation of Algorithm 3 we assume that K is a field, R is a K-algebra, $x \in R$ and ι is an algebraic inverse function on $K[x]$. Then f is non-constant and $f(x) = 0$, i.e. x is algebraic over K.*

Algorithm 4. *Let a discrete field structure K, a discrete K-algebra structure R, two elements $x_1, x_2 \in R$ and $\iota : K[X_1, X_2] \to K[X_1, X_2]$ be given. We compute $f_1, f_2 \in K[X]$ as follows starting with f_1:*

1. *If $x_2 = 0$, we use Algorithm 3 with input K, R, $x_1 \in R$ and $\iota' : K[X] \to K[X]$ defined by $\iota'(p) := \iota(p(X_1))(X, 0)$ and return the output as f_1.*
2. *Otherwise, compute $\iota(X_2)$ and define g as the polynomial which comes from $X_2\iota(X_2)-1 \in K[X_1, X_2]$ by dropping each coefficient $p \in K[X_1]$ with $p(x_1) = 0$ and let $h \in K[X_1]$ be the leading coefficient of g (and 1 if $g = 0$).*

3. *Apply Algorithm 2 to the input* K, $\vec{h} := (X_1, \iota(h))$, ι, $g_0^{(1)} = Y_1$ *and* $g_{k_2-1}^{(2)}, \dots, g_0^{(2)} \in K[Y_1, Y_2]$ *are the coefficients of* $g(Y_1, X)$*, except the leading coefficient, multiplied with* Y_2*. Let* $\tilde{\iota} : K[Y_1, Y_2] \to K[Y_1, Y_2]$ *be the output of this algorithm.*
4. *If* $\deg(h) = 0$*, (i.e.* $h = h_0$ *for some* $h_0 \in K$*), apply Algorithm 3 to* K*,* R*,* $x_1 \in R$ *and* $\iota' : K[X] \to K[X]$ *given by* $\iota'(p) := \tilde{\iota}(p(Y_1))(X, h_0^{-1})$ *and return the output of this algorithm as* f_1*.*
5. *Otherwise, check if* $1 - h(x_1) = 0$*. If yes, return* $f_1 := 1 - h(X)$*.*
6. *If no, compute* $\tilde{\iota}(1 - h(Y_1)) = \sum_{i=0}^{N} a_i Y_2^i$ *with* $a_i \in K[Y_1]$ *and* $a_N \neq 0$*; define*

$$q := \sum_{i=0}^{N} a_i (h(Y_1))^{N-i} \in K[Y_1]$$

and return $f_1 := h(X)^N - (1 - h(X))q(X)$*.*

Change x_1 *and* x_2 *and repeat the steps above to compute* $f_2 \in K[X]$*.*

Lemma 6. *In the situation of Algorithm 4 we assume that* K *is a field,* R *is a* K*-algebra,* ι *is an algebraic inverse function on* $K[x_1, x_2]$*. Then* $f_1(x_1) = f_2(x_2) = 0$ *and* f_1, f_2 *are non-constant.*

The next algorithm shows how to compute the field L, which corresponds to the field of fractions of $K[x_1]$ in $K[\vec{x}]$ on the level of polynomials.

Algorithm 5. *Let a discrete field structure* K*, a discrete* K*-algebra structure* R*,* $n > 0$ *and* $\vec{x} \in R$ *be given. We define a field structure as follows:*

$$L := \left\{ \frac{f}{g} \middle| f, g \in K[X], g(x_1) \neq 0 \vee 0 = 1 \right\},$$

$$\frac{f_1}{g_1} = \frac{f_2}{g_2} :\Leftrightarrow f_1(x_1)g_2(x_1) = f_2(x_1)g_1(x_1),$$

$$\frac{f_1}{g_1} + \frac{f_2}{g_2} := \begin{cases} \frac{f_1 g_2 + f_2 g_1}{g_1 g_2} & \text{if } (g_1 g_2)(x_1) \neq 0 \\ \frac{0}{1} & \text{else,} \end{cases} \qquad 0 := \frac{0}{1},$$

$$\frac{f_1}{g_1}\frac{f_2}{g_2} := \begin{cases} \frac{f_1 f_2}{g_1 g_2} & \text{if } (g_1 g_2)(x_1) \neq 0 \\ \frac{0}{1} & \text{else,} \end{cases} \qquad 1 := \frac{1}{1},$$

$$-\frac{f}{g} := \frac{-f}{g}, \qquad \left(\frac{f}{g}\right)^{-1} := \begin{cases} \frac{g}{f} & \text{if } f(x_1) \neq 0 \\ \frac{0}{1} & \text{else} \end{cases}$$

For a given map $\iota : K[\vec{X}] \to K[\vec{X}]$ *we define a map* $\varphi : L \to R$ *by* $\frac{f}{g} \mapsto f(x_1)(\iota(g))(x_1)$*, which turns* R *into an* L*-algebra structure. Furthermore, we define a map* $\tilde{\iota} : L[X_2, \dots, X_n] \to L[X_2, \dots, X_n]$ *as follows:*

1. *Given an input* $p \in L[X_2, \dots, X_n]$*, it has the presentation*

$$p = \sum_{i_2, \dots, i_n} \frac{f_{i_2 \dots i_n}}{g_{i_2 \dots i_n}} X_2^{i_2} \cdots X_n^{i_n},$$

for finitely many $f_{i_2\dots i_n}, g_{i_2\dots i_n} \in K[X]$.
2. *Let* $a \in K[X]$ *be the product of all these* $g_{i_2\dots i_n}$*, and for* $j_2, \dots, j_n$ *let* $h_{j_2\dots j_n}$ *be the product of all these* $g_{i_2\dots i_n}$ *except* $g_{j_2\dots j_n}$.
3. *Define* $\tilde{f}_{i_2\dots i_n} := f_{i_2\dots i_n} h_{i_2\dots i_n}$ *and* $\tilde{p} := \sum_{i_2,\dots,i_n} \tilde{f}_{i_2\dots i_n}(X_1) X_2^{i_2} \cdots X_n^{i_n}$*; set*

$$\tilde{\iota}(p) := (a(X_1)\iota(\tilde{p}))\left(\frac{X}{1}, X_2, \dots, X_n\right),$$

where we consider $b \in K$ *also as the element* $\frac{b}{1} \in L$.

Because we have to define the algorithm without the ring and field axioms, the definitions of L and the operators are more complex than one might expect.

As already mentioned we cannot define L as $\left\{\frac{a}{b} \mid a, b \in K[x_1], b \neq 0 \vee 0 = 1\right\}$, which is the field of fractions of $K[x_1]$ if this is an integral domain, because we want to avoid terms like $a \in K[x_1]$, which are constructively delicate. In particular, there is in general no map which takes $a \in K[x_1]$ and returns $f \in K[X]$ with $f(x_1) = a$ without using the axiom of choice. But in the next algorithm we operate on the level of polynomials.

Lemma 7. *In the situation of Algorithm 5 we assume that* K *is a field,* R *is a ring,* $\vec{x} \in R$ *and* ι *is an algebraic inverse function of* $K[\vec{x}]$*. Then* L *is indeed a discrete field,* φ *turns* R *into a* L*-algebra and* $\tilde{\iota}$ *is an algebraic inverse function on* $L[x_2, \dots, x_n]$.

With this preparation we now formulate the final algorithm and an algorithm version of Zariski's lemma.

Algorithm 6. *Let* K *be a discrete field structure,* R *be a discrete* K*-algebra structure,* $\iota : K[\vec{X}] \to K[\vec{X}]$ *be a map and* $x_1, \dots, x_n \in R$*. We compute* $f_1, \dots, f_n \in K[X]$ *by recursion over* n *as follows:*

1. *If* $n = 0$*, return the empty list. If* $n = 1$*, use Algorithm 3 with input* K*,* R*,* x_1 *and* ι *and return the output* f_1*. If* $n = 2$*, use Algorithm 4 with input* K*;* R*;* $x_1, x_2 \in R$ *and* ι*, and return the output* f_1, f_2.
2. *Apply Algorithm 5 to* K*,* R*,* n*,* $\vec{x}$ *and* ι *and let the field structure* L *and the map* $\iota' : L[X_2, \dots, X_n] \to L[X_2, \dots, X_n]$ *be the output.*
3. *Apply recursion to* L*, the* L*-algebra structure* R*,* ι' *and* $x_2, \dots, x_n \in R$ *and we get* $\tilde{F}_2, \dots, \tilde{F}_n \in L[X]$.
4. *For each* i *we define* F_i *as* $\tilde{F}_i$ *divided by its leading coefficient and replacing the leading coefficient by* 1 *(or* $F_i := 1$ *if* $\tilde{F}_i = 0$*). In particular,*

$$F_i = X^{n_i} + \sum_{j=0}^{n_i - 1} \frac{a_{ij}}{b_{ij}} X^j$$

for some $a_{ij}, b_{ij} \in K[X]$.
5. *Let* $v := \prod_{(k,l)} b_{kl} \in K[X]$*,* $\tilde{b}_{ij} := \prod_{(k,l)\neq(i,j)} b_{kl}$*, and* $\tilde{a}_{ij} := \tilde{b}_{ij} a_{ij}$*. Define*

$$G_i := \sum_{j=0}^{n_i} \tilde{a}_{ij}(Y_1) Y_2 X^j \in K[Y_1, Y_2, X].$$

6. *Use Algorithm 2 with input* K, $\vec{h} := (X_1, \iota(v))$, ι, $k_1 := 1$, $g_0^{(1)} := Y_1$ *and for* $i \in \{2, \dots, n\}$ *take* $k_i := n_i$ *and* $g_{n_i-1}^{(i)}, \dots, g_0^{(i)}$ *are the non-leading coefficients of* G_i. *Let* $\tilde{\iota}$ *be the output.*
7. *Apply Algorithm 4 to the input* K, R, $x_1, \iota(v)(x_1) \in R$ *and* $\tilde{\iota}$, *and define* $f_1 \in K[X]$ *as the output.*
8. *For each* $i \in \{2, \dots, n\}$ *exchange* x_1 *with* x_i *and repeat the processes starting at Step 2 to get* f_i *instead of* f_1. *Then return* $f_1, \dots, f_n$.

Theorem 2 (Algorithmic version of Zariski's lemma). *In the situation of Algorithm 6 we assume that* K *is a field,* R *is a* K*-algebra,* $\vec{x} \in R$ *and* ι *is an algebraic inverse function on* $K[\vec{x}]$. *Then* $f_1(x_1) = \cdots = f_n(x_n) = 0$ *and* $f_1, \dots, f_n$ *are non-constant.*

4 Summary and Outlook

For $K[x_1, \dots, x_n]$ and an algebraic inverse function ι on $K[\vec{x}]$ our algorithm computes $f_1, \dots, f_n$ with $f_i(x_i) = 0$ for all i as follows: The case $n = 0$ is trivial. The case $n = 1$ is given in Lemma 5. The algorithm uses now recursion on n and reduction to the case $n = 2$. The case $n = 2$ itself is considered in Lemma 6. In this lemma the main idea was to find a suitable element u such that $K[x_1, u] \subseteq K[x_1, x_2]$ is an integral extension of K-algebras. By using Lemma 4 we have an algebraic inverse function on $K[x_1, u]$, where Lemma 4 uses Lemma 3. In the case $n \geq 3$, we use Lemma 7 to produce a new field L over which the original algebra is generated by one element less, such that we can use recursion and get $F_2, \dots, F_n \in L[X]$ with $F_i(x_i) = 0$ for all i. From these F_i's we generate v such that $K[x_1, v] \subseteq K[\vec{x}]$ is an integral extension of K-algebras. Using again Lemma 4 we get an algebraic inverse function on $K[x_1, v]$ and therefore, again by Lemma 6, we get $f_1 \in K[X]$ with $f_1(x_1) = 0$. One now repeats the algorithm where x_1 and x_i are switched for all $i \geq 2$ and get $f_i \in K[X]$ with $f_i(x_i) = 0$.

Using the theory given in [11, 12] one can probably formulate an algorithmic version of Hilbert's Nullstellensatz if the underlying field is countable. Another direction in which this paper can be extended is an analysis of the complexity of the algorithm. The algorithm of Section 3 as a whole is defined by recursion over the number of generators. In the recursion step (i.e. Algorithm 6) the algorithm with input $x_1, \dots, x_n$ relies on the algorithm with input $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n$ for each $i \leq n$. Therefore, the runtime of this algorithm must be at least quadratic in the number of generators.

A Omitted proofs

Proof (Lemma 3). We define the $K[\vec{Y}]$-module $M := K[\vec{Y}][\vec{X}]/\langle G_1, \ldots, G_n\rangle$ where $G_i := X^{k_i} + g^{(i)}_{k_i-1}X^{k_i-1} + \cdots + g^{(i)}_0$ for all i, and go through the steps of Algorithm 1: by the definition of M and the process to get the g_{IJ}'s, we have

$$\sum_{J\in\mathbb{N}^n} f_{IJ}\vec{X}^J = \sum_{J\in\mathcal{I}} g_{IJ}(\vec{Y})\vec{X}^J$$

in M or in other words

$$\sum_{J\in\mathbb{N}^n} f_{IJ}\vec{X}^J - \sum_{J\in\mathcal{I}} g_{IJ}(\vec{Y})\vec{X}^J \in \langle G_1, \ldots, G_n\rangle$$

seen in $K[\vec{Y}][\vec{X}]$. Note that $(\vec{X}^I)_{I\in\mathcal{I}}$ is a set of generators of M as $K[\vec{Y}]$-module, and multiplication with f corresponds to the matrix $(g_{IJ})_{I,J\in\mathcal{I}}$. Let P be the characteristic polynomial as in the algorithm. By the theorem of Cayley-Hamilton [4], $P(f) = 0$ in M, hence $P(f) \in \langle G_1, \ldots, G_n\rangle$ in $K[\vec{Y}][\vec{X}]$. By (1), we have $G_i(\vec{y}, \vec{x}) = 0$ for all i, and hence $0 = P(f)(\vec{y}, \vec{x}) = (f(\vec{x}))^k + g_{k-1}(\vec{y})(f(\vec{x}))^{k-1} + \cdots + g_0(\vec{y})$. Here we have used the definition of the g_i in the last step, and $\deg(P) = k$ because k is the number of elements in $\mathcal{I}$ which is also the cardinality of the generator $(x^I)_{I\in\mathcal{I}}$. □

Proof (Lemma 4). Let $f \in K[\vec{Y}]$ with $f(\vec{y}) \neq 0$ be given. Since $\vec{h}$ is a witness that $K[\vec{y}] \subseteq K[\vec{x}]$ is an extension of K-algebras, we have $f(\vec{h}(\vec{x})) = f(\vec{y}) \neq 0$. Let p be given as in Step 1. Then $p(\vec{x})f(\vec{y}) = 1$ because ι is an algebraic inverse function. By Lemma 3 we have

$$(p(\vec{x}))^k + g_{k-1}(\vec{y})(p(\vec{x}))^{k-1} + \cdots + g_0(\vec{y}) = 0.$$

Multiplying this with $(f(\vec{y}))^{k-1}$ and isolating $p(\vec{x})$, we get

$$p(\vec{x}) = (-g_{k-1} - g_{k-2}f - \cdots - g_0 f^{k-1})(\vec{y}) = \tilde{\iota}(f)(\vec{y}).$$

□

The proof of Lemma 5 follows directly by the definition of an algebraic inverse function.

Proof (Lemma 6). It suffices to consider f_1 since the statement with f_2 is proved analogously. We follow the algorithm step by step. If $x_2 = 0$, we use Lemma 5. That ι' is an algebraic inverse function on $K[x_1]$ follows from

$$(\iota(p(X_1)))(x_1, 0)p(x_1) = (\iota(p(X_1))p(X_1))(x_1, x_2) = 1$$

for all $p \in K[X]$ with $p(x_1) \neq 0$.

So, we continue with $x_2 \neq 0$. By definition, $g(x_1, x_2) = x_2\iota(X_2)(x_1, x_2) - 1 = 0$ and the constant coefficient (as polynomial in X_2) of g is equal to -1.

In the next step it is obvious that $X_1, \iota(h)$ is a witness of $K[x_1, \iota(h)(x_1, x_2)] \subseteq K[x_1, x_2]$ being an extension of K-algebras and that $x_1 - g_0^{(1)}(x_1, \iota(h)(x_1, x_2)) = x_1 - x_1 = 0$. Furthermore, let $g = \sum_{i=0}^{k_2} g_i X_2^i$ for some $g_i \in K[X_1]$ with $g_{k_2} \neq 0$. Then $h = g_{k_2}$ and

$$\begin{aligned} 0 = \iota(h)(x_1, x_2) g(x_1, x_2) &= x_2^{k_2} + \sum_{i=0}^{k_2-1} g_i(x_1) \iota(h)(x_1, x_2) x_2^i \\ &= x_2^{k_2} + \sum_{i=0}^{k_2-1} g_i^{(2)}(x_1, \iota(h)(x_1, x_2)) x_2^i. \end{aligned}$$

So, $\tilde{\iota}$ is an algebraic inverse function on $K[x_1, \iota(h)(x_1, x_2)]$ by Lemma 4.

If $\deg(h) = 0$, we have $h = h_0$ and $h_0 \neq 0$ because h is a leading coefficient. Therefore, it follows $\iota(h)(x_1, x_2) = h_0^{-1}$, and we apply Lemma 5 to $K[x_1] = K[x_1, h_0^{-1}]$. To apply this lemma it remains to show that ι' is an algebraic inverse function: if $p \in K[X]$ with $p(x_1) \neq 0$ then

$$(\iota'(p))(x_1) p(x_1) = (\tilde{\iota}(p(Y_1)))(x_1, h_0^{-1}) p(x_1) = (\tilde{\iota}(p(Y_1)) p(Y_1))(x_1, h_0^{-1}) = 1.$$

Now we continue with $\deg(h) \neq 0$, i.e. $\deg(h) > 0$ because $h \neq 0$. If $h(x_1) + 1 = 0$, we have that f_1 is non-constant since $\deg(h) > 0$ and by the case assumption $f(x_1) = 0$.

So let $h(x_1) + 1 \neq 0$. Then

$$q(x_1)(\iota(h)(x_1, x_2))^N = \tilde{\iota}(1 - h(Y_1))(x_1, \iota(h)(x_1, x_2)).$$

Since ι and $\tilde{\iota}$ are algebraic inverse functions and $h \neq 0$ and $1 - h \neq 0$, it follows

$$q(x_1)(1 - h(x_1))) = h(x_1).$$

So for $f_1 := (1 - h(X))q(X) - h(X)^N$ we have $f_1(x_1) = 0$ and $f_1 \neq 0$, similar to the end of the proof of Zariski's lemma. □

Proof (Lemma 7). L is a discrete field because in the definition of L and its operators we only use the operators of K.

By using the property of an algebraic inverse function, it is also straightforward to check that the map φ is a homomorphism.

It remains to show that $\tilde{\iota}$ is an algebraic inverse function on $L[x_2, \ldots, x_n]$. For this let $p \in L[X_2, \ldots, X_n]$ with $p(x_2, \ldots, x_n) \neq 0$ be given. We take the representation of p, a, $\tilde{f}_{i_2 \ldots i_n}$ and $\tilde{p}$ as defined in the algorithm, and calculate

$$\begin{aligned} p(x_2 \ldots, x_n) a(x_1) &= \sum_{i_2, \ldots, i_n} \varphi\left(\frac{\tilde{f}_{i_2 \ldots i_n}}{1} \right) x_2^{i_2} \cdots x_n^{i_n} \\ &= \sum_{i_2, \ldots, i_n} \tilde{f}_{i_2 \ldots i_n}(x_1) x_2^{i_2} \cdots x_n^{i_n} = \tilde{p}(x_1, \ldots, x_n). \end{aligned}$$

Obviously, $a(x_1) \neq 0$ because it is a product of non-zero factors. Hence, if $p(x_2, \ldots, x_n) \neq 0$, it follows $\tilde{p}(x_1, \ldots x_n) \neq 0$. Since additionally ι is an algebraic inverse function, we have

$$\iota(\tilde{p})(x_1, \ldots, x_n) = (\tilde{p}(x_1, \ldots, x_n))^{-1},$$

and therefore

$$\begin{aligned}(p(x_2, \ldots, x_n))^{-1} &= a(x_1)(\tilde{p}(x_1, \ldots, x_n))^{-1} = a(x_1)(\iota(\tilde{p}))(x_1, \ldots, x_n) \\ &= (a(X_1)\iota(\tilde{p}))\left(\frac{X}{1}, x_2, \ldots, x_n\right).\end{aligned}$$

□

Proof (Algorithmic version of Zariski's Lemma). We use induction on n and consider the algorithm step by step. If $n = 0$, there is nothing to show. If $n = 1$, the statement follows by Lemma 5. If $n = 2$, the statement follows by Lemma 6.

If $n \geq 3$, it suffices to consider $e = 1$. We use Lemma 7 to get that L is a field, R is an L-algebra and ι' is an algebraic inverse function on $L[x_2, \ldots, x_n]$.

We have that $F_2(x_2) = \cdots = F_n(x_n) = 0$ by the induction hypothesis and the fact that F_i is indeed $\tilde{F}_i$ divided by its leading coefficient since L is a field.

Furthermore, $F_i = G_i(x_1, v^{-1}, X)$ as polynomial in $R[X]$ and therefore $0 = F_i(x_i) = G_i(x_1, (v(x_1))^{-1}, x_i)$. So, the non-leading coefficients of G_i (as polynomials in X) witness that x_i is integral over $K[x_1, \iota(v)(x_1)]$ for each $i \in \{2, \ldots, n\}$.

Because of this, the requirements of Lemma 4 are fulfilled and hence $\tilde{\iota}$ is an algebraic inverse function on $K[x_1, \iota(v)(x_1)]$.

Therefore, we get $f_1(x_1) = 0$ and f_1 is non-constant by Lemma 6. □

References

1. Atiyah, M.F., Macdonald, I.G.: Introduction to Commutative Algebra. Addison-Wesley Pub. Co., Boston (1969)
2. Azarang, A.: A simple proof of Zariski's Lemma. Bulletin of the Iranian Mathematical Society **43**(5), 1529–1530 (2017)
3. Berger, U., Miyamoto, K., Schwichtenberg, H., Seisenberger, M.: Minlog - A Tool for Program Extraction Supporting Algebras and Coalgebra. In: Interna. Conference on Algebra and Coalgebra in Computer Science. pp. 393–399. Springer (2011)
4. Eisenbud, D.: Commutative Algebra: with a View Toward Algebraic Geometry, Graduate Texts in Mathematics, vol. 150. Springer Science & Business Media (1995)
5. Hulek, K.: Elementare algebraische Geometrie: grundlegende Begriffe und Techniken mit zahlreichen Beispielen und Anwendungen. Springer-Verlag (2012)
6. Kohlenbach, U.: Applied Proof Theory: Proof Interpretations and their Use in Mathematics. Springer Science & Business Media (2008)
7. Kohlenbach, U.: Proof-theoretic methods in nonlinear anlysis. In: Proc. of the International Congress of Mathematicians. vol. 2, pp. 61–82. World Scientific (2018)
8. Lombardi, H., Quitté, C.: Commutative Algebra: Constructive Methods. Springer, Berlin (2015)

9. McCabe, J.: A Note on Zariski's Lemma. The American Mathematical Monthly **83**(7), 560–561 (1976)
10. Mines, R., Richman, F., Ruitenburg, W.: A Course in Constructive Algebra. Springer Science & Business Media (1988)
11. Powell, T., Schuster, P., Wiesnet, F.: An algorithmic approach to the existence of ideal objects in commutative algebra. In: International Workshop on Logic, Language, Information, and Computation. pp. 533–549. Springer (2019)
12. Powell, T., Schuster, P., Wiesnet, F.: A universal algorithm for Krull's theorem (2020), submitted
13. Reid, M.: Undergraduate Algebraic Geometry. Cambridge University Press Cambridge (1988)
14. Schwichtenberg, H., Wainer, S.S.: Proofs and Computations. Cambridge University Press (2011)
15. Sharifi, Y.: Zariski's Lemma. https://ysharifi.wordpress.com/tag/zariskis-lemma/ [Acccessed: 7 June 2020] (2011)
16. Wessel, D.: Making the use of maximal ideals inductive (2021), talk at workshop *Reducing complexity in algebra, logic, combinatorics*
17. Wiesnet, F.: Introduction to Minlog. In: Mainzer, K., Schuster, P., Schwichtenberg, H. (eds.) Proof and Computation. pp. 233–288. World Scientific (2018)
18. Yengui, I.: Constructive Commutative Algebra. Projective Modules over Polynomial Rings and Dynamical Gröbner Bases, Lecture Notes in Mathematics, vol. 2138. Springer, Cham (2015)
19. Zariski, O.: A new proof of Hilbert's Nullstellensatz. Bulletin of the American Mathematical Society **53**(4), 362–368 (1947)