

# Alon’s Combinatorial Nullstellensatz

Arthur F. Ramos

David Barros Hulak

Ruy J. G. B. de Queiroz

May 10, 2026

## Abstract

This entry formalizes Alon’s Combinatorial Nullstellensatz for sparse multivariate polynomials over fields. The proof derives the coefficient formula from univariate Lagrange interpolation and uses it to obtain the standard nonvanishing conclusion over finite grids. AI assistance was used for proof engineering. The final definitions, statements, and proofs are checked by Isabelle.

## 1 Overview

Alon’s Combinatorial Nullstellensatz is a polynomial-method theorem about a distinguished coefficient of a multivariate polynomial. In the form formalized here, if all monomials have total degree bounded by  $d_1 + \dots + d_n$  and the coefficient of  $X_1^{d_1} \dots X_n^{d_n}$  is nonzero, then the polynomial has a nonzero value on every product grid whose  $i$ -th side has more than  $d_i$  elements.

The Isabelle/HOL development represents multivariate polynomials sparsely, as finite-support coefficient functions from exponent lists to field elements. It first proves the necessary univariate Lagrange interpolation identities, then sums polynomial evaluations over a finite grid with the corresponding Lagrange weights. All monomials except the distinguished exponent vector cancel, yielding the coefficient formula and the nonvanishing theorem.

## 2 Sources

The entry follows Alon’s original paper [1]. The formal proof uses only Isabelle/HOL’s standard computational algebra library as its mathematical base.

## Contents

<b>1 Overview</b>	<b>1</b>
<b>2 Sources</b>	<b>1</b>
<b>3 Alon’s Combinatorial Nullstellensatz</b>	<b>1</b>
3.1 Univariate interpolation . . . . .	2
3.2 Sparse multivariate polynomials . . . . .	3
<b>theory</b> <i>Combinatorial-Nullstellensatz</i>	
<b>imports</b> <i>HOL-Computational-Algebra.Polynomial</i>	
<b>begin</b>	

## 3 Alon’s Combinatorial Nullstellensatz

Alon’s Combinatorial Nullstellensatz [1] is a polynomial method theorem: if the coefficient of a distinguished monomial of total degree  $d_1 + \dots + d_n$  is nonzero, then the polynomial cannot vanish on every point of any grid whose  $i$ -th side has more than  $d_i$  elements.

The development below proves this statement for sparse multivariate polynomials over arbitrary fields. A polynomial is represented by a finite-support coefficient function from exponent lists to coefficients. This keeps the formalization independent of a particular multivariate polynomial library while still matching the usual mathematical statement.

### 3.1 Univariate interpolation

The proof starts with a small amount of univariate interpolation. For a finite set  $S$  and a point  $x \in S$ , the following denominator and basis polynomial are the standard Lagrange factors.

**definition** *lagrange-denom* :: 'a::field set  $\Rightarrow$  'a  $\Rightarrow$  'a **where**  
*lagrange-denom*  $S$   $x = (\prod_{y \in S - \{x\}}. x - y)$

**definition** *lagrange-basis* :: 'a::field set  $\Rightarrow$  'a  $\Rightarrow$  'a **poly where**  
*lagrange-basis*  $S$   $x =$   
 $smult$  (*inverse* (*lagrange-denom*  $S$   $x$ ))  
 $(\prod_{y \in S - \{x\}}. [- y, 1:])$

**lemma** *lagrange-denom-nonzero*:  
**assumes** *finite*  $S$   $x \in S$   
**shows** *lagrange-denom*  $S$   $x \neq 0$   
 $\langle$ *proof* $\rangle$

**lemma** *poly-lagrange-basis*:  
**assumes** *finite*  $S$   $x \in S$   $z \in S$   
**shows** *poly* (*lagrange-basis*  $S$   $x$ )  $z =$  (*if*  $z = x$  *then* 1 *else* 0)  
 $\langle$ *proof* $\rangle$

**lemma** *degree-lagrange-basis-le*:  
**assumes** *finite*  $S$   $x \in S$   
**shows** *degree* (*lagrange-basis*  $S$   $x$ )  $\leq$  *card*  $S - 1$   
 $\langle$ *proof* $\rangle$

**lemma** *lagrange-interpolation*:  
**fixes**  $P ::$  'a::{field,ring-no-zero-divisors} **poly**  
**assumes** *fin*: *finite*  $S$  **and** *deg*: *degree*  $P <$  *card*  $S$   
**shows**  $P = (\sum_{x \in S}. smult$  (*poly*  $P$   $x$ ) (*lagrange-basis*  $S$   $x$ ))  
 $\langle$ *proof* $\rangle$

**lemma** *coeff-lagrange-basis-top*:  
**assumes** *fin*: *finite*  $S$  **and**  $x: x \in S$   
**shows** *poly.coeff* (*lagrange-basis*  $S$   $x$ ) (*card*  $S - 1$ ) =  
*inverse* (*lagrange-denom*  $S$   $x$ )  
 $\langle$ *proof* $\rangle$

**lemma** *lagrange-power-sum*:  
**fixes**  $S ::$  'a::field **set**  
**assumes** *fin*: *finite*  $S$  **and** *card*: *card*  $S = Suc$   $d$   
**assumes**  $k: k \leq d$   
**shows**  $(\sum_{x \in S}. x ^ k /$  *lagrange-denom*  $S$   $x$ ) = (*if*  $k = d$  *then* 1 *else* 0)  
 $\langle$ *proof* $\rangle$

**lemma** *lagrange-power-sum-list*:  
**fixes**  $xs ::$  'a::field **list**  
**assumes** *dist*: *distinct*  $xs$  **and** *len*: *length*  $xs = Suc$   $d$  **and**  $k: k \leq d$   
**shows** *sum-list* (*map*  $(\lambda x. x ^ k /$  *lagrange-denom* (*set*  $xs$ )  $x$ )  $xs$ ) = (*if*  $k = d$  *then* 1 *else* 0)  
 $\langle$ *proof* $\rangle$

## 3.2 Sparse multivariate polynomials

Monomials are indexed by exponent lists. The value of  $[e_1, \dots, e_n]$  at  $[x_1, \dots, x_n]$  is  $x_1^{\wedge e_1} * \dots * x_n^{\wedge e_n}$ ; mismatched lengths evaluate to zero. The predicate *sparse-poly* records finite support and a fixed arity.

**fun** *monomial-value* :: *nat list*  $\Rightarrow$  *'a::comm-semiring-1 list*  $\Rightarrow$  *'a* **where**  
*monomial-value* [] [] = 1  
| *monomial-value* (e # es) (x # xs) = x  $\wedge$  e \* *monomial-value* es xs  
| *monomial-value* - - = 0

**fun** *grid-weight* :: *'a::field list list*  $\Rightarrow$  *'a list*  $\Rightarrow$  *'a* **where**  
*grid-weight* [] [] = 1  
| *grid-weight* (S # Ss) (x # xs) = *lagrange-denom* (set S) x \* *grid-weight* Ss xs  
| *grid-weight* - - = 1

**definition** *support* :: (*nat list*  $\Rightarrow$  *'a::zero*)  $\Rightarrow$  *nat list set* **where**  
*support* p = {m. p m  $\neq$  0}

**definition** *sparse-poly* :: *nat*  $\Rightarrow$  (*nat list*  $\Rightarrow$  *'a::zero*)  $\Rightarrow$  *bool* **where**  
*sparse-poly* n p  $\iff$  *finite* (*support* p)  $\wedge$  ( $\forall$  m  $\in$  *support* p. *length* m = n)

**definition** *total-degree-le* :: (*nat list*  $\Rightarrow$  *'a::zero*)  $\Rightarrow$  *nat*  $\Rightarrow$  *bool* **where**  
*total-degree-le* p d  $\iff$  ( $\forall$  m  $\in$  *support* p. *sum-list* m  $\leq$  d)

**definition** *eval-sparse-poly* :: (*nat list*  $\Rightarrow$  *'a::comm-semiring-1*)  $\Rightarrow$  *'a list*  $\Rightarrow$  *'a* **where**  
*eval-sparse-poly* p xs = ( $\sum$  m  $\in$  *support* p. p m \* *monomial-value* m xs)

**lemma** *product-lists-set-Cons*:

*set* (*product-lists* (xs # xss)) = ( $\lambda$ (x, ys). x # ys) ‘ (*set* xs  $\times$  *set* (*product-lists* xss))  
<proof>

**lemma** *sum-list-concat*:

*sum-list* (*concat* xss) = *sum-list* (*map* *sum-list* xss)  
<proof>

**lemma** *sum-list-map-zero*:

**fixes** f :: *'a*  $\Rightarrow$  *'b::monoid-add*  
**assumes**  $\bigwedge$ x. x  $\in$  *set* xs  $\implies$  f x = 0  
**shows** *sum-list* (*map* f xs) = 0  
<proof>

**lemma** *sum-list-product-lists-Cons*:

*sum-list* (*map* f (*product-lists* (xs # xss))) =  
*sum-list* (*map* ( $\lambda$ x. *sum-list* (*map* ( $\lambda$ ys. f (x # ys)) (*product-lists* xss))) xs)  
<proof>

**lemma** *grid-weight-Cons*:

*grid-weight* (S # Ss) (x # xs) = *lagrange-denom* (set S) x \* *grid-weight* Ss xs  
<proof>

**lemma** *monomial-value-Cons*:

*monomial-value* (e # es) (x # xs) = x  $\wedge$  e \* *monomial-value* es xs  
<proof>

**definition** *grid-monom-sum* :: *'a::field list list*  $\Rightarrow$  *nat list*  $\Rightarrow$  *'a* **where**

*grid-monom-sum* Xss es =  
*sum-list* (*map* ( $\lambda$ xs. *monomial-value* es xs / *grid-weight* Xss xs) (*product-lists* Xss))

**lemma** *grid-monom-sum-Cons*:  
**assumes** *dist*: *distinct Xs*  
**shows**  $\text{grid-monom-sum } (Xs \# Xss) (e \# es) =$   
 $\text{sum-list } (\text{map } (\lambda x. x \wedge e / \text{lagrange-denom } (\text{set } Xs) x) Xs) *$   
 $\text{grid-monom-sum } Xss es$   
 $\langle \text{proof} \rangle$

**lemma** *grid-monom-sum-Nil* [*simp*]:  
 $\text{grid-monom-sum } [] [] = 1$   
 $\langle \text{proof} \rangle$

**lemma** *monomial-value-wrong-length*:  
 $\text{length } es \neq \text{length } xs \implies \text{monomial-value } es xs = 0$   
 $\langle \text{proof} \rangle$

**lemma** *grid-monom-sum-wrong-length*:  
**assumes**  $\text{length } es \neq \text{length } Xss$   
**shows**  $\text{grid-monom-sum } Xss es = 0$   
 $\langle \text{proof} \rangle$

**lemma** *grid-monom-sum-delta*:  
**fixes**  $Xss :: 'a::\text{field list list}$   
**assumes** *grids*:  $\text{list-all2 } (\lambda Xs d. \text{distinct } Xs \wedge \text{length } Xs = \text{Suc } d) Xss ds$   
**assumes** *len*:  $\text{length } es = \text{length } ds$   
**assumes** *deg*:  $\text{sum-list } es \leq \text{sum-list } ds$   
**shows**  $\text{grid-monom-sum } Xss es = (\text{if } es = ds \text{ then } 1 \text{ else } 0)$   
 $\langle \text{proof} \rangle$

The next lemma is the coefficient formula specialized to the sparse representation: the weighted sum of all grid evaluations extracts exactly the coefficient of the target exponent list.

**lemma** *sum-list-sum*:  
**fixes**  $f :: 'b \Rightarrow 'c \Rightarrow 'a::\text{comm-monoid-add}$   
**assumes** *finite*  $A$   
**shows**  $\text{sum-list } (\text{map } (\lambda x. \sum a \in A. f x a) xs) =$   
 $(\sum a \in A. \text{sum-list } (\text{map } (\lambda x. f x a) xs))$   
 $\langle \text{proof} \rangle$

**lemma** *eval-sparse-poly-grid-sum*:  
**fixes**  $p :: \text{nat list} \Rightarrow 'a::\text{field}$   
**assumes** *sp*: *sparse-poly* ( $\text{length } ds$ )  $p$   
**assumes** *grids*:  $\text{list-all2 } (\lambda Xs d. \text{distinct } Xs \wedge \text{length } Xs = \text{Suc } d) Xss ds$   
**assumes** *deg*: *total-degree-le*  $p$  ( $\text{sum-list } ds$ )  
**shows**  $\text{sum-list } (\text{map } (\lambda xs. \text{eval-sparse-poly } p xs / \text{grid-weight } Xss xs) (\text{product-lists } Xss)) =$   
 $p ds$   
 $\langle \text{proof} \rangle$

**theorem** *combinatorial-nullstellensatz-exact-lists*:  
**fixes**  $p :: \text{nat list} \Rightarrow 'a::\text{field}$   
**assumes** *sp*: *sparse-poly* ( $\text{length } ds$ )  $p$   
**assumes** *deg*: *total-degree-le*  $p$  ( $\text{sum-list } ds$ )  
**assumes** *coeff*:  $p ds \neq 0$   
**assumes** *grids*:  $\text{list-all2 } (\lambda Xs d. \text{distinct } Xs \wedge \text{length } Xs = \text{Suc } d) Xss ds$   
**shows**  $\exists xs \in \text{set } (\text{product-lists } Xss). \text{eval-sparse-poly } p xs \neq 0$   
 $\langle \text{proof} \rangle$

Finally, the standard “more than  $d_i$  points” formulation follows by selecting  $d_i + 1$  points from each side of the grid.

**definition** *exact-grid-sublists* ::  $'a \text{ list list} \Rightarrow \text{nat list} \Rightarrow 'a \text{ list list}$  **where**  
 $\text{exact-grid-sublists } Xss ds = \text{map } (\lambda (Xs, d). \text{take } (\text{Suc } d) Xs) (\text{zip } Xss ds)$

**lemma** *exact-grid-sublists-all2*:

**assumes** *list-all2* ( $\lambda Xs\ d.$  *distinct*  $Xs \wedge$  *length*  $Xs > d$ )  $Xss\ ds$

**shows** *list-all2* ( $\lambda Xs\ d.$  *distinct*  $Xs \wedge$  *length*  $Xs = \text{Suc } d$ )

(*exact-grid-sublists*  $Xss\ ds$ )  $ds$

*<proof>*

**lemma** *exact-grid-sublists-subset*:

**assumes** *list-all2* ( $\lambda Xs\ d.$  *length*  $Xs > d$ )  $Xss\ ds$

**shows** *set* (*product-lists* (*exact-grid-sublists*  $Xss\ ds$ ))  $\subseteq$  *set* (*product-lists*  $Xss$ )

*<proof>*

**theorem** *combinatorial-nullstellensatz-lists*:

**fixes**  $p :: \text{nat list} \Rightarrow 'a::\text{field}$

**assumes** *sp*: *sparse-poly* (*length*  $ds$ )  $p$

**assumes** *deg*: *total-degree-le*  $p$  (*sum-list*  $ds$ )

**assumes** *coeff*:  $p\ ds \neq 0$

**assumes** *grids*: *list-all2* ( $\lambda Xs\ d.$  *distinct*  $Xs \wedge$  *length*  $Xs > d$ )  $Xss\ ds$

**shows**  $\exists xs \in \text{set} (\text{product-lists } Xss). \text{eval-sparse-poly } p\ xs \neq 0$

*<proof>*

**end**

## References

- [1] N. Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8(1–2):7–29, 1999. DOI: <https://doi.org/10.1017/S0963548398003411>.