Decartes' Rule of Signs in Lean

Will Dey

Abstract. A large formalization project of Decartes' celebrated Rule of Signs is started in the Lean theorem prover. The theorem is one of the few classical results frequently taught in undergraduate mathematics that has yet to be formalized in Lean. In the course of the formalization, many important and general lemmas regarding polynomials over rings and their representations as lists of coefficients are proven.

1 Introduction

Throughout the last century of mathematics, there as been an increasing focus on rigor in order to ensure absence of paradoxes and correctness of complex proofs. However, the point at which a proof meets the modern standard of rigor had itself not been standardized and rigorously defined for many decades, and instead relied on the collective agreement of a critical mass of mathematicians. With the recent advent of computers, digital theorem provers have presented an answer to this dilemma: by formalizing a mathematical proof in a theorem proving language which only accepts proofs that can be mechanically checked from axioms, a proof can be guaranteed to be consistent in both a universal way, and one that does not need convincing of error-prone humans.

Lean is one such interactive digital theorem prover. It has seen recent adoption in the Xena project, which aims to formalize all of undergraduate mathematics in it, as well as graduate mathematics proofs such as the Liquid Tensor Experiment and the proof of the existence of Sphere Eversions—both of which are highly nontrivial and advanced proofs. Despite the breadth of results proven so far in Lean, many gaps remain. Dr. Freek Wiedijk maintains a list of 100 well-known theorems, which are arbitrarily chosen, but nonetheless

span a wide range of mathematical subfields and is conventionally used to compare different theorem provers and the relative ease or power of each one in proving statements that are widespread in unformalized mathematics. Of this list, only 24 remain to be proven in Lean.

I have chosen to prove Decartes' Rule of Signs for my final project, because

- it is one of the 24 theorems that have never been proven in Lean yet,
- it "sparse polynomial" representation is an relatively uncommon proof technique in mathematics,
- it is an important part of undergraduate mathematics,
- and it is currently in widespread practical use today for fast numerical rootfinding of polynomials in engineering.

2 Proof

Theorem. If the nonzero terms of a single-variable polynomial with real coefficients are ordered by descending exponent of the variable, then the number of positive roots of the polynomial is either equal to the number of sign changes between consecutive nonzero coefficients, or is less than it by an even number. (For example, if the number of sign changes is 0 or 1, then there are exactly 0 or 1 positive roots, respectively.)

Proof: I follow the modern elementary proof of Xiaoshen https://www.jstor.org/stable/4145072?origin=crossref. Write the polynomial $f(x)$ in the sparse representation as

$$\sum_{i=0}^{n} a_i x^{b_i},$$

where we have integer powers $0 < g_0 < b_1 < \cdots < b_n$ and nonzero coefficients $a_i \neq 0$. Let $V(f)$ be the number of sign changes of the coefficients of f, meaning the number of k such that $a_k a_{k+1} < 0$. Let $Z(f)$ be the number of strictly positive roots, counting multiplicity. Using these definitions, in order to proof the theorem, it suffices to prove that the number of strictly positive roots (counting multiplicity) of f is equal to the number of sign changes in the coefficients of f, minus a nonnegative even number.

If $b_0 > 0$, then we can divide the polynomial by b_0, which would not change its number of strictly positive roots. Without loss of generality, let $b_0 = 0$.

Lemma. If $a_n a_0 > 0$, then $Z(f)$ is even. If $a_n a_0 < 0$, then $Z(f)$ is odd. (This concludes all cases because $a_n a_0 \neq 0$ because $a_n, a_0 \neq 0$ by definition.)

Proof: $f(x)$ starts at $f(0) = a_0 > 0$ and ends at $f(+\infty) = +\infty > 0$, so it must cross the positive x-axis an even number of times, each of which contributes to an odd number of roots, and glance without cross the positive x-axis an arbitrary number of times (each of which contributes an even number of roots). The case for the other part is done similarly in Lean. ■

From the lemma, it follows that $Z(f)$ and $V(f)$ always have the same parity. We now show that $Z(f) \leq V(f)$. We induct on n. If $n = 0, 1$, the proof is straightforward, and the example was given previously. Assume $n \geq 2$. By the Induction Hypothesis, $Z(f') = V(f') - 2s$ for some $s \in \mathbb{Z}^+$. By Rolle's theorem, there exists at least one positive root of f' between any two different positive roots of f. Also, any k-multiple positive root of f is a $k-1$-multiple root of f'. Therefore, $Z(f') \geq Z(f) - 1$. ■

If $a_0 a_1 \geq 0$, then $V(f') = V(f)$, else $V(f') = V(f) - 1$. In either case, $V(f') \leq V(f)$.

Combining these gives us

$$Z(f) \leq Z(f') + 1 = V(f') - 2s + 1 \leq V(f) - 2s + 1 \leq V(f) + 1$$

Further, since $Z(f)$ and $V(f)$ have the same parity, we have $Z(f) \leq V(f)$. □

3 Lean and `mathlib`

I have used descriptive names in place of $V(f)$ and $Z(f)$ in my formalization of the Theorem. In particular, $Z(f)$ is defined as

```
def positive_roots (f : ℝ[X]) : multiset ℝ :=
  multiset.filter (λ a, 0 < a) (polynomial.roots f)
```

showing the dependence on `mathlib`'s `data.polynomial.basic` for `polynomial.roots`.

As mentioned previously, since the sparse representation of polynomials, or "fewnomials", is uncommon, my Lean definition of $V(f)$

```
def sign_changes (f : ℝ[X]) : ℕ :=
  (list.destutter (≠)
   $ list.map (real.sign ∘ f.coeff)
   $ finset.sort (≤) f.support).length - 1
```

first uses the `finset` of the powers with nonzero coefficients of f `f.support`, converts it to an ordered list, retrieves the corresponding coefficient signs, and then makes heavy use of `mathlib`'s convenient `list.destutter`, which deduplicates only consecutive duplicates. In this way, I have formalized the intuitive notion of counting of sign changes in Lean.

Because `list.destutter` itself is uncommon in `mathlib`, I proved two utility lemmas on its interaction with other `list` operations in order to make the destuttered representation useful for the proving of Decartes' Rule of Signs:

- `list.destutter'_map`
- `list.destutter_map`

Other lemmas I have made to flesh out `mathlib`'s offerings (in a completely general way, so that they may be used by other projects), include:

- `polynomial.support_C` (mirroring the existing `polynomial.coeff_C`)
- `polynomial.trailing_coeff_neg` (mirroring the existing `polynomial.leading_coeff_neg`)
- `polynomial.trailing_coeff_mul_X` (mirroring the existing `polynomial.leading_coeff_mul_X`)
- `polynomial.erase_lead_trailing_coeff`
- `polynomial.div_mul`
- `polynomial.div_by_monic_mul_pow_root_multiplicity_roots`
- `real.sign_pos_iff`
- `real.sign_neg_iff`

I have also added a new definition `div_all_X` which repeatedly divides out factors of x from a polynomial (used in my application of the `mathlib` `wlog` tactic to prove the Lemma in a general way), and corresponding lemmas on it:

- `coeff_div_all_X`
- `leading_coeff_div_all_X`
- `nat_trailing_degree_div_all_X`
- `trailing_coeff_div_all_X`

Finally, since existing `induction ... using` proved too rigid to prove the Lemma on roots, I defined a new inductive type to eliminate `Prop`s in `mathlib`'s `induction` tactic in two steps rather than one:

```
theorem list.even_induction {α : Type} {P : list α  Prop}
  (hnil : P list.nil)
  (hone : ∀ {a : α}, P [a])
  (htwo : ∀ {a b : α}, P [a, b])
  (hrest : ∀ {a b : α} {tl : list α}, P tl  P (a :: b :: tl))
  : ∀ l, P l
| [] := hnil
| [a] := hone
| [a, b] := htwo
| (a :: b :: tl) := hrest (list.even_induction tl)
```

To make my formalization modular, I have formalized the theorem as

$$Z(f) \equiv V(f) \pmod 2$$

$$\wedge$$

$$Z(f) \leq V(f),$$

written in Lean naturally as

```
theorem decartes_rule_of_signs (f : ℝ[X])
  : (positive_roots f).card ≡ sign_changes f [MOD 2]
  ∧ (positive_roots f).card ≤ sign_changes f
```

This statement is equivalent to the Theorem and lends itself to separation of the two parts of the inductive datatype ∧, which are individually proved as `decartes_rule_of_signs''` and `decartes_rule_of_signs'`, respectively. The Lemma is only used in proving `decartes_rule_of_signs''` and is itself broken into two parts for the even and odd cases, since even though the two cases are intuitively similar, the details of formalization for even multiplicities and odd multiplicities of roots is drastically different. The two cases are

```
lemma card_positive_roots_of_pos_leading_trailing {f : ℝ[X]}
  (h : 0 < f.leading_coeff * f.trailing_coeff) : even (positive_roots f).card
```

and

```
lemma card_positive_roots_of_neg_leading_trailing {f : ℝ[X]}
  (h : f.leading_coeff * f.trailing_coeff < 0) : odd (positive_roots f).card
```

respectively.

My overall code is 476 lines of Lean long, and though it still contains some instances of `sorry` due to time constraints, it represents significant progress towards formalizing this important theorem in Lean.