The Budan-Fourier Theorem and Counting Real Roots with Multiplicity

Wenda Li

December 14, 2021

Abstract

This entry is mainly about counting and approximating real roots (of a polynomial) with multiplicity. We have first formalised the Budan-Fourier theorem: given a polynomial with real coefficients, we can calculate sign variations on Fourier sequences to over-approximate the number of real roots (counting multiplicity) within an interval. When all roots are known to be real, the over-approximation becomes tight: we can utilise this theorem to count real roots exactly. It is also worth noting that Descartes' rule of sign is a direct consequence of the Budan-Fourier theorem, and has been included in this entry. In addition, we have extended previous formalised Sturm's theorem to count real roots with multiplicity, while the original Sturm's theorem only counts distinct real roots. Compared to the Budan-Fourier theorem, our extended Sturm's theorem always counts roots exactly but may suffer from greater computational cost.

Many problems in real algebraic geometry is about counting or approximating roots of a polynomial. Previous formalised results are mainly about counting distinct real roots (i.e. Sturm's theorem in Isabelle/HOL [5, 2], HOL Light [4], PVS [9] and Coq [8]) and limited support for multiple real roots (i.e. Descartes' rule of signs in Isabelle/HOL [3], HOL Light and ProofPower[1]). In comparison, this entry provides more comprehensive support for reasoning about multiple real roots.

The main motivation of this entry is to cope with the roots-on-the-border issue when counting complex roots [7, 6], but the results here should be beneficial to other developments.

Our proof of the Budan-Fourier theorem mainly follows Theorem 2.35 in the book by Basu et al. [1] and that of the extended Sturm's theorem is inspired by Theorem 10.5.6 in Rahman and Schmeisser's book [10].

[1]According to Freek Wiedijk's "Formalising 100 Theorems" (http://www.cs.ru.nl/~freek/100/index.html)

1 Misc results for polynomials and sign variations

theory *BF-Misc* **imports**
HOL−Computational-Algebra.Polynomial-Factorial
HOL−Computational-Algebra.Fundamental-Theorem-Algebra
Sturm-Tarski.Sturm-Tarski
begin

1.1 Induction on polynomial roots

lemma *poly-root-induct-alt* [*case-names 0 no-proots root*]:
fixes p :: $'a$ :: *idom poly*
assumes $Q\ 0$
assumes $\bigwedge p.\ (\bigwedge a.\ poly\ p\ a \neq 0) \Longrightarrow Q\ p$
assumes $\bigwedge a\ p.\ Q\ p \Longrightarrow Q\ ([:-a,\ 1:] * p)$
shows $Q\ p$
⟨*proof*⟩

1.2 Misc

lemma *lead-coeff-pderiv*:
fixes p :: $'a$::{*comm-semiring-1*,*semiring-no-zero-divisors*,*semiring-char-0*} *poly*
shows *lead-coeff* (*pderiv p*) = *of-nat* (*degree p*) ∗ *lead-coeff p*
⟨*proof*⟩

lemma *gcd-degree-le-min*:
assumes $p \neq 0$ $q \neq 0$
shows *degree* (*gcd p q*) ≤ *min* (*degree p*) (*degree q*)
⟨*proof*⟩

lemma *lead-coeff-normalize-field*:
fixes p::$'a$::{*field*,*semidom-divide-unit-factor*} *poly*
assumes $p \neq 0$
shows *lead-coeff* (*normalize p*) = *1*
⟨*proof*⟩

lemma *smult-normalize-field-eq*:
fixes p::$'a$::{*field*,*semidom-divide-unit-factor*} *poly*
shows p = *smult* (*lead-coeff p*) (*normalize p*)
⟨*proof*⟩

lemma *lead-coeff-gcd-field*:
fixes $p\ q$::$'a$::*field-gcd poly*
assumes $p \neq 0 \lor q \neq 0$
shows *lead-coeff* (*gcd p q*) = *1*
⟨*proof*⟩

lemma *poly-gcd-0-iff*:
$poly\ (gcd\ p\ q)\ x = 0 \longleftrightarrow poly\ p\ x{=}0 \land poly\ q\ x{=}0$
⟨*proof*⟩

lemma *degree-eq-oneE*:
  fixes p :: $'a$::*zero poly*
  assumes *degree* $p = 1$
  obtains a b **where** $p = [:a,b:]$ $b \neq 0$
⟨*proof*⟩

1.3 More results about sign variations (i.e. *changes*

lemma *changes-0*[*simp*]:*changes* ($0\#xs$) = *changes* xs
  ⟨*proof*⟩

lemma *changes-Cons*:*changes* ($x\#xs$) = (*if filter* ($\lambda x.\ x\neq 0$) xs = [] *then*
      0
    else if $x*$ *hd* (*filter* ($\lambda x.\ x\neq 0$) xs) < 0 *then*
      $1 +$ *changes* xs
    else changes xs)
  ⟨*proof*⟩

lemma *changes-filter-eq*:
  changes (*filter* ($\lambda x.\ x\neq 0$) xs) = *changes* xs
  ⟨*proof*⟩

lemma *changes-filter-empty*:
  assumes *filter* ($\lambda x.\ x\neq 0$) xs = []
  shows *changes* $xs = 0$ *changes* ($a\#xs$) $= 0$ ⟨*proof*⟩

lemma *changes-append*:
  assumes $xs\neq$ [] $\wedge$ $ys\neq$ [] $\longrightarrow$ (*last* xs = *hd* ys $\wedge$ *last* $xs\neq 0$)
  shows *changes* ($xs@ys$) = *changes* xs + *changes* ys
  ⟨*proof*⟩

lemma *changes-drop-dup*:
  assumes $xs\neq$ [] $ys\neq$ [] $\longrightarrow$ *last* xs=*hd* ys
  shows *changes* ($xs@ys$) = *changes* ($xs@$ *tl* ys)
  ⟨*proof*⟩

lemma *Im-poly-of-real*:
  Im (*poly* p (*of-real* x)) = *poly* (*map-poly Im* p) x
  ⟨*proof*⟩

lemma *Re-poly-of-real*:
  Re (*poly* p (*of-real* x)) = *poly* (*map-poly Re* p) x
  ⟨*proof*⟩

1.4 More about *map-poly* and *of-real*

lemma *of-real-poly-map-pCons*[*simp*]:*map-poly of-real* (*pCons a p*) = *pCons* (*of-real a*) (*map-poly of-real p*)
⟨*proof*⟩

lemma *of-real-poly-map-plus*[*simp*]: *map-poly of-real* (*p* + *q*) = *map-poly of-real p* + *map-poly of-real q*
⟨*proof*⟩

lemma *of-real-poly-map-smult*[*simp*]:*map-poly of-real* (*smult s p*) = *smult* (*of-real s*) (*map-poly of-real p*)
⟨*proof*⟩

lemma *of-real-poly-map-mult*[*simp*]:*map-poly of-real* (*p*∗*q*) = *map-poly of-real p* ∗ *map-poly of-real q*
⟨*proof*⟩

lemma *of-real-poly-map-poly*:
of-real (*poly p x*) = *poly* (*map-poly of-real p*) (*of-real x*)
⟨*proof*⟩

lemma *of-real-poly-map-power*:*map-poly of-real* (*p*^*n*) = (*map-poly of-real p*) ^ *n*
⟨*proof*⟩

lemma *of-real-poly-eq-iff* [*simp*]: *map-poly of-real p* = *map-poly of-real q* ⟷ *p* = *q*
⟨*proof*⟩

lemma *of-real-poly-eq-0-iff* [*simp*]: *map-poly of-real p* = *0* ⟷ *p* = *0*
⟨*proof*⟩

1.5 More about *order*

lemma *order-multiplicity-eq*:
assumes *p*≠*0*
shows *order a p* = *multiplicity* [:−*a*,*1*:] *p*
⟨*proof*⟩

lemma *order-gcd*:
assumes *p*≠*0* *q*≠*0*
shows *order x* (*gcd p q*) = *min* (*order x p*) (*order x q*)
⟨*proof*⟩

lemma *order-linear*[*simp*]: *order x* [:−*a*,*1*:] = (*if x*=*a then 1 else 0*)
⟨*proof*⟩

lemma *map-poly-order-of-real*:

assumes $p{\neq}0$
shows $order\ (of\text{-}real\ t)\ (map\text{-}poly\ of\text{-}real\ p) = order\ t\ p$ ⟨*proof*⟩

lemma *order-pcompose*:
assumes $pcompose\ p\ q{\neq}0$
shows $order\ x\ (pcompose\ p\ q) = order\ x\ (q-[:poly\ q\ x:]) * order\ (poly\ q\ x)\ p$
⟨*proof*⟩

1.6 Polynomial roots / zeros

definition $proots\text{-}within::'a::comm\text{-}semiring\text{-}0\ poly \Rightarrow\ 'a\ set \Rightarrow\ 'a\ set$ **where**
$proots\text{-}within\ p\ s = \{x{\in}s.\ poly\ p\ x{=}0\}$

abbreviation $proots::'a::comm\text{-}semiring\text{-}0\ poly \Rightarrow\ 'a\ set$ **where**
$proots\ p \equiv proots\text{-}within\ p\ UNIV$

lemma *proots-def*: $proots\ p = \{x.\ poly\ p\ x{=}0\}$
⟨*proof*⟩

lemma *proots-within-empty*[*simp*]:
$proots\text{-}within\ p\ \{\} = \{\}$ ⟨*proof*⟩

lemma *proots-within-0*[*simp*]:
$proots\text{-}within\ 0\ s = s$ ⟨*proof*⟩

lemma *proots-withinI*[*intro*,*simp*]:
$poly\ p\ x{=}0 \Longrightarrow x{\in}s \Longrightarrow x{\in}proots\text{-}within\ p\ s$
⟨*proof*⟩

lemma *proots-within-iff*[*simp*]:
$x{\in}proots\text{-}within\ p\ s \longleftrightarrow poly\ p\ x{=}0 \wedge x{\in}s$
⟨*proof*⟩

lemma *proots-within-union*:
$proots\text{-}within\ p\ A \cup proots\text{-}within\ p\ B = proots\text{-}within\ p\ (A \cup B)$
⟨*proof*⟩

lemma *proots-within-times*:
fixes $s::'a::\{semiring\text{-}no\text{-}zero\text{-}divisors,comm\text{-}semiring\text{-}0\}\ set$
shows $proots\text{-}within\ (p{*}q)\ s = proots\text{-}within\ p\ s \cup proots\text{-}within\ q\ s$
⟨*proof*⟩

lemma *proots-within-gcd*:
fixes $s::'a::\{factorial\text{-}ring\text{-}gcd,semiring\text{-}gcd\text{-}mult\text{-}normalize\}\ set$
shows $proots\text{-}within\ (gcd\ p\ q)\ s{=}\ proots\text{-}within\ p\ s \cap proots\text{-}within\ q\ s$
⟨*proof*⟩

lemma *proots-within-inter*:
$NO\text{-}MATCH\ UNIV\ s \Longrightarrow proots\text{-}within\ p\ s = proots\ p \cap s$

$\langle proof \rangle$

lemma *proots-within-proots*[*simp*]:
proots-within p s $\subseteq$ *proots p*
$\langle proof \rangle$

lemma *finite-proots*[*simp*]:
fixes *p* :: *'a::idom poly*
shows $p \neq 0 \Longrightarrow$ *finite* (*proots-within p s*)
$\langle proof \rangle$

lemma *proots-within-pCons-1-iff*:
fixes *a::'a::idom*
shows *proots-within* [:$-a$,*1*:] *s* = (*if* $a \in s$ *then* $\{a\}$ *else* $\{\}$)
proots-within [:*a*,-1:] *s* = (*if* $a \in s$ *then* $\{a\}$ *else* $\{\}$)
$\langle proof \rangle$

lemma *proots-within-uminus*[*simp*]:
fixes *p* :: *'a::comm-ring poly*
shows *proots-within* ($-$ *p*) *s* = *proots-within p s*
$\langle proof \rangle$

lemma *proots-within-smult*:
fixes *a::'a::*{*semiring-no-zero-divisors*,*comm-semiring-0*}
assumes $a \neq 0$
shows *proots-within* (*smult a p*) *s* = *proots-within p s*
$\langle proof \rangle$

1.7 Polynomial roots counting multiplicities.

definition *proots-count::'a::idom poly* $\Rightarrow$ *'a set* $\Rightarrow$ *nat* **where**
proots-count p s = ($\sum$ $r \in$ *proots-within p s*. *order r p*)

lemma *proots-count-emtpy*[*simp*]:*proots-count p* {} = *0*
$\langle proof \rangle$

lemma *proots-count-times*:
fixes *s* :: *'a::idom set*
assumes $p*q \neq 0$
shows *proots-count* ($p*q$) *s* = *proots-count p s* + *proots-count q s*
$\langle proof \rangle$

lemma *proots-count-power-n-n*:
shows *proots-count* ([:$-$ *a*, *1*:]^*n*) *s* = (*if* $a \in s \wedge n>0$ *then n else 0*)
$\langle proof \rangle$

lemma *degree-proots-count*:
fixes *p::complex poly*
shows *degree p* = *proots-count p UNIV*

⟨*proof*⟩

lemma *proots-count-smult*:
fixes a::$'a$::{*semiring-no-zero-divisors*,*idom*}
assumes $a \neq 0$
shows *proots-count* (*smult a p*) s= *proots-count p s*
⟨*proof*⟩

lemma *proots-count-pCons-1-iff*:
fixes a::$'a$::*idom*
shows *proots-count* [:$-a$,*1*:] s = (*if* $a \in s$ *then 1 else 0*)
⟨*proof*⟩

lemma *proots-count-uminus*[*simp*]:
proots-count ($-$ *p*) *s* = *proots-count p s*
⟨*proof*⟩

lemma *card-proots-within-leq*:
assumes $p \neq 0$
shows *proots-count p s* $\geq$ *card* (*proots-within p s*) ⟨*proof*⟩

lemma *proots-count-0-imp-empty*:
assumes *proots-count p s*=*0* $p \neq 0$
shows *proots-within p s* = {}
⟨*proof*⟩

lemma *proots-count-leq-degree*:
assumes $p \neq 0$
shows *proots-count p s*$\leq$ *degree p* ⟨*proof*⟩

lemma *proots-count-union-disjoint*:
assumes $A \cap B = \{\}$ $p \neq 0$
shows *proots-count p* ($A \cup B$) = *proots-count p A* + *proots-count p B*
⟨*proof*⟩

lemma *proots-count-cong*:
assumes *order-eq*:$\forall x \in s$. *order x p* = *order x q* **and** $p \neq 0$ **and** $q \neq 0$
shows *proots-count p s* = *proots-count q s* ⟨*proof*⟩

lemma *proots-count-of-real*:
assumes $p \neq 0$
shows *proots-count* (*map-poly of-real p*) ((*of-real*::-$\Rightarrow$$'a$::{*real-algebra-1*,*idom*}) ‘ *s*)
= *proots-count p s*
⟨*proof*⟩

lemma *proots-pcompose*:
  fixes *p q*::*'a*::*field poly*
  assumes $p \neq 0$ *degree* $q=1$
  shows *proots-count* (*pcompose p q*) *s* = *proots-count p* (*poly q ‘ s*)
⟨*proof*⟩

1.8 Composition of a polynomial and a rational function

definition *fcompose*::*'a* ::*field poly* ⇒ *'a poly* ⇒ *'a poly* ⇒ *'a poly* **where**
  fcompose p q r = *fst* (*fold-coeffs* (λ*a* (*c*,*d*). (*d*∗[:*a*:] + *q* ∗ *c*,*r*∗*d*)) *p* (*0*,*1*))

lemma *fcompose-0* [*simp*]: *fcompose 0 q r* = *0*
  ⟨*proof*⟩

lemma *fcompose-const*[*simp*]:*fcompose* [:*a*:] *q r* = [:*a*:]
  ⟨*proof*⟩

lemma *fcompose-pCons*:
  fcompose (*pCons a p*) *q1 q2* = *smult a* (*q2*^(*degree* (*pCons a p*))) + *q1* ∗ *fcompose p q1 q2*
⟨*proof*⟩

lemma *fcompose-uminus*:
  fcompose (−*p*) *q r* = − *fcompose p q r*
  ⟨*proof*⟩

lemma *fcompose-add-less*:
  assumes *degree p1* > *degree p2*
  shows *fcompose* (*p1*+*p2*) *q1 q2*
      = *fcompose p1 q1 q2* + *q2*^(*degree p1*−*degree p2*) ∗ *fcompose p2 q1 q2*
  ⟨*proof*⟩

lemma *fcompose-add-eq*:
  assumes *degree p1* = *degree p2*
  shows *q2*^(*degree p1* − *degree* (*p1*+*p2*)) ∗ *fcompose* (*p1*+*p2*) *q1 q2*
      = *fcompose p1 q1 q2* + *fcompose p2 q1 q2*
  ⟨*proof*⟩

lemma *fcompose-add-const*:
  fcompose ([:*a*:] + *p*) *q1 q2* = *smult a* (*q2* ^ *degree p*) + *fcompose p q1 q2*
  ⟨*proof*⟩

lemma *fcompose-smult*: *fcompose* (*smult a p*) *q1 q2* = *smult a* (*fcompose p q1 q2*)
  ⟨*proof*⟩

lemma *fcompose-mult*: *fcompose* (*p1*∗*p2*) *q1 q2* = *fcompose p1 q1 q2* ∗ *fcompose p2 q1 q2*
⟨*proof*⟩

lemma *fcompose-poly*:
  assumes *poly q2 x≠0*
  shows *poly p* (*poly q1 x*/*poly q2 x*) = *poly* (*fcompose p q1 q2*) *x* / *poly* (*q2*^(*degree p*)) *x*
  ⟨*proof*⟩

lemma *poly-fcompose*:
  assumes *poly q2 x≠0*
  shows *poly* (*fcompose p q1 q2*) *x* = *poly p* (*poly q1 x*/*poly q2 x*) ∗ (*poly q2 x*)^(*degree p*)
  ⟨*proof*⟩
lemma *poly-fcompose-0-denominator*:
  assumes *poly q2 x=0*
  shows *poly* (*fcompose p q1 q2*) *x* = *poly q1 x* ^ *degree p* ∗ *lead-coeff p*
  ⟨*proof*⟩

lemma *fcompose-0-denominator*:*fcompose p q1 0* = *smult* (*lead-coeff p*) (*q1*^*degree p*)
  ⟨*proof*⟩

lemma *fcompose-nzero*:
  fixes *p*::′*a*::*field poly*
  assumes *p≠0* **and** *q2≠0* **and** *nconst*:∀ *c*. *q1* ≠ *smult c q2*
    and *infi*:*infinite* (*UNIV*::′*a set*)
  shows *fcompose p q1 q2* ≠ *0* ⟨*proof*⟩

1.9 Bijection (*bij-betw*) and the number of polynomial roots

lemma *proots-fcompose-bij-eq*:
  fixes *p*::′*a*::*field poly*
  assumes *bij*:*bij-betw* (λ*x*. *poly q1 x*/*poly q2 x*) *A B* **and** *p≠0*
    and *nzero*:∀ *x*∈*A*. *poly q2 x≠0*
    and *max-deg*: *max* (*degree q1*) (*degree q2*) ≤ *1*
    and *nconst*:∀ *c*. *q1* ≠ *smult c q2*
    and *infi*:*infinite* (*UNIV*::′*a set*)
  shows *proots-count p B* = *proots-count* (*fcompose p q1 q2*) *A*
  ⟨*proof*⟩

lemma *proots-card-fcompose-bij-eq*:
  fixes *p*::′*a*::*field poly*
  assumes *bij*:*bij-betw* (λ*x*. *poly q1 x*/*poly q2 x*) *A B* **and** *p≠0*
    and *nzero*:∀ *x*∈*A*. *poly q2 x≠0*
    and *max-deg*: *max* (*degree q1*) (*degree q2*) ≤ *1*
    and *nconst*:∀ *c*. *q1* ≠ *smult c q2*
    and *infi*:*infinite* (*UNIV*::′*a set*)
  shows *card* (*proots-within p B*) = *card* (*proots-within* (*fcompose p q1 q2*) *A*)
  ⟨*proof*⟩

lemma *proots-pcompose-bij-eq*:
fixes p::$'a$::*idom poly*
assumes *bij*:*bij-betw* (λx. *poly q x*) *A B* **and** $p \neq 0$
and *q-deg*: *degree q* = *1*
shows *proots-count p B* = *proots-count* ($p \circ_p q$) *A* ⟨*proof*⟩

lemma *proots-card-pcompose-bij-eq*:
fixes p::$'a$::*idom poly*
assumes *bij*:*bij-betw* (λx. *poly q x*) *A B* **and** $p \neq 0$
and *q-deg*: *degree q* = *1*
shows *card* (*proots-within p B*) = *card* (*proots-within* ($p \circ_p q$) *A*) ⟨*proof*⟩

end

2 Budan-Fourier theorem

theory *Budan-Fourier* **imports**
BF-Misc
begin

The Budan-Fourier theorem is a classic result in real algebraic geometry to over-approximate real roots of a polynomial (counting multiplicity) within an interval. When all roots of the the polynomial are known to be real, the over-approximation becomes tight – the number of roots are counted exactly. Also note that Descartes' rule of sign is a direct consequence of the Budan-Fourier theorem.

The proof mainly follows Theorem 2.35 in Basu, S., Pollack, R., Roy, M.-F.: Algorithms in Real Algebraic Geometry. Springer Berlin Heidelberg, Berlin, Heidelberg (2006).

2.1 More results related to *sign-r-pos*

lemma *sign-r-pos-nzero-right*:
assumes *nzero*:$\forall x.\ c<x \wedge x\leq d \longrightarrow$ *poly p x* $\neq 0$ **and** $c<d$
shows *if sign-r-pos p c then poly p d*>*0 else poly p d*<*0*
⟨*proof*⟩

lemma *sign-r-pos-at-left*:
assumes $p \neq 0$
shows *if even* (*order c p*) $\longleftrightarrow$*sign-r-pos p c then eventually* (λx. *poly p x*>*0*) (*at-left c*)
else eventually (λx. *poly p x*<*0*) (*at-left c*)
⟨*proof*⟩

lemma *sign-r-pos-nzero-left*:
assumes *nzero*:$\forall x.\ d\leq x \wedge x<c \longrightarrow$ *poly p x* $\neq 0$ **and** $d<c$
shows *if even* (*order c p*) $\longleftrightarrow$*sign-r-pos p c then poly p d*>*0 else poly p d*<*0*
⟨*proof*⟩

2.2 Fourier sequences

function *pders*::*real poly* ⇒ *real poly list* **where**
pders p = (*if p* =*0 then* [] *else Cons p* (*pders* (*pderiv p*)))
⟨*proof*⟩
termination
⟨*proof*⟩

declare *pders.simps*[*simp del*]

lemma *set-pders-nzero*:
assumes *p*≠*0 q*∈*set* (*pders p*)
shows *q*≠*0*
⟨*proof*⟩

2.3 Sign variations for Fourier sequences

definition *changes-itv-der*:: *real* ⇒ *real* ⇒*real poly* ⇒ *int* **where**
changes-itv-der a b p= (*let ps*= *pders p in changes-poly-at ps a* − *changes-poly-at ps b*)

definition *changes-gt-der*:: *real* ⇒*real poly* ⇒ *int* **where**
changes-gt-der a p= *changes-poly-at* (*pders p*) *a*

definition *changes-le-der*:: *real* ⇒*real poly* ⇒ *int* **where**
changes-le-der b p= (*degree p* − *changes-poly-at* (*pders p*) *b*)

lemma *changes-poly-pos-inf-pders*[*simp*]:*changes-poly-pos-inf* (*pders p*) = *0*
⟨*proof*⟩

lemma *changes-poly-neg-inf-pders*[*simp*]: *changes-poly-neg-inf* (*pders p*) = *degree p*
⟨*proof*⟩

lemma *pders-coeffs-sgn-eq*:*map* (*λp. sgn*(*poly p 0*)) (*pders p*) = *map sgn* (*coeffs p*)
⟨*proof*⟩

lemma *changes-poly-at-pders-0*:*changes-poly-at* (*pders p*) *0* = *changes* (*coeffs p*)
⟨*proof*⟩

2.4 Budan-Fourier theorem

lemma *budan-fourier-aux-right*:
assumes *c*<*d2* **and** *p*≠*0*
assumes ∀ *x. c*<*x*∧ *x*≤*d2* ⟶ (∀ *q*∈*set* (*pders p*). *poly q x*≠*0*)
shows *changes-itv-der c d2 p*=*0*
⟨*proof*⟩

lemma *budan-fourier-aux-left'*:
assumes *d1*<*c* **and** *p*≠*0*

assumes $\forall x.\ d1 \leq x \wedge x < c \longrightarrow (\forall q \in set\ (pders\ p).\ poly\ q\ x \neq 0)$
shows $changes\text{-}itv\text{-}der\ d1\ c\ p \geq order\ c\ p \wedge even\ (changes\text{-}itv\text{-}der\ d1\ c\ p - order\ c\ p)$
⟨*proof*⟩

lemma *budan-fourier-aux-left*:
assumes $d1 < c$ **and** $p \neq 0$
assumes *nzero*:$\forall x.\ d1 < x \wedge x < c \longrightarrow (\forall q \in set\ (pders\ p).\ poly\ q\ x \neq 0)$
shows $changes\text{-}itv\text{-}der\ d1\ c\ p \geq order\ c\ p\ even\ (changes\text{-}itv\text{-}der\ d1\ c\ p - order\ c\ p)$
⟨*proof*⟩

theorem *budan-fourier-interval*:
assumes $a < b\ p \neq 0$
shows $changes\text{-}itv\text{-}der\ a\ b\ p \geq proots\text{-}count\ p\ \{x.\ a < x \wedge x \leq b\} \wedge$
$even\ (changes\text{-}itv\text{-}der\ a\ b\ p - proots\text{-}count\ p\ \{x.\ a < x \wedge x \leq b\})$
⟨*proof*⟩

theorem *budan-fourier-gt*:
assumes $p \neq 0$
shows $changes\text{-}gt\text{-}der\ a\ p \geq proots\text{-}count\ p\ \{x.\ a < x\} \wedge$
$even\ (changes\text{-}gt\text{-}der\ a\ p - proots\text{-}count\ p\ \{x.\ a < x\})$
⟨*proof*⟩

Descartes' rule of signs is a direct consequence of the Budan-Fourier theorem

theorem *descartes-sign*:
fixes *p*::*real poly*
assumes $p \neq 0$
shows $changes\ (coeffs\ p) \geq proots\text{-}count\ p\ \{x.\ 0 < x\} \wedge$
$even\ (changes\ (coeffs\ p) - proots\text{-}count\ p\ \{x.\ 0 < x\})$
⟨*proof*⟩

theorem *budan-fourier-le*:
assumes $p \neq 0$
shows $changes\text{-}le\text{-}der\ b\ p \geq proots\text{-}count\ p\ \{x.\ x \leq b\} \wedge$
$even\ (changes\text{-}le\text{-}der\ b\ p - proots\text{-}count\ p\ \{x.\ x \leq b\})$
⟨*proof*⟩

2.5 Count exactly when all roots are real

definition *all-roots-real*:: *real poly* ⇒ *bool* **where**
$all\text{-}roots\text{-}real\ p = (\forall r \in proots\ (map\text{-}poly\ of\text{-}real\ p).\ Im\ r = 0)$

lemma *all-roots-real-mult*[*simp*]:
$all\text{-}roots\text{-}real\ (p * q) \longleftrightarrow all\text{-}roots\text{-}real\ p \wedge all\text{-}roots\text{-}real\ q$
⟨*proof*⟩

lemma *all-roots-real-const-iff*:

assumes *all-real*:*all-roots-real p*
shows *degree* $p \neq 0 \longleftrightarrow (\exists\, x.\ poly\ p\ x{=}0)$
⟨*proof*⟩

lemma *all-roots-real-degree*:
assumes *all-roots-real p*
shows *proots-count p UNIV =degree p* ⟨*proof*⟩

lemma *all-real-roots-mobius*:
fixes *a b*::*real*
assumes *all-roots-real p* **and** $a<b$
shows *all-roots-real* (*fcompose p* [:*a*,*b*:] [:*1*,*1*:]) ⟨*proof*⟩

If all roots are real, we can use the Budan-Fourier theorem to EXACTLY count the number of real roots.

corollary *budan-fourier-real*:
assumes $p \neq 0$
assumes *all-roots-real p*
shows *proots-count p* $\{x.\ x \leq a\}$ = *changes-le-der a p*
$a<b \Longrightarrow$ *proots-count p* $\{x.\ a < x \wedge x \leq b\}$ = *changes-itv-der a b p*
proots-count p $\{x.\ b < x\}$ = *changes-gt-der b p*
⟨*proof*⟩

Similarly, Descartes' rule of sign counts exactly when all roots are real.

corollary *descartes-sign-real*:
fixes *p*::*real poly* **and** *a b*::*real*
assumes $p \neq 0$
assumes *all-roots-real p*
shows *proots-count p* $\{x.\ 0 < x\}$ = *changes* (*coeffs p*)
⟨*proof*⟩

end

3 Extension of Sturm's theorem for multiple roots

theory *Sturm-Multiple-Roots*
imports
BF-Misc
begin

The classic Sturm's theorem is used to count real roots WITHOUT multiplicity of a polynomial within an interval. Surprisingly, we can also extend Sturm's theorem to count real roots WITH multiplicity by modifying the signed remainder sequence, which seems to be overlooked by many textbooks.

Our formal proof is inspired by Theorem 10.5.6 in Rahman, Q.I., Schmeisser, G.: Analytic Theory of Polynomials. Oxford University Press (2002).

3.1 More results for *smods*

lemma *last-smods-gcd*:
  fixes *p q* ::*real poly*
  defines *pp* ≡ *last* (*smods p q*)
  assumes *p*≠*0*
  shows *pp* = *smult* (*lead-coeff pp*) (*gcd p q*)
  ⟨*proof*⟩

lemma *last-smods-nzero*:
  assumes *p*≠*0*
  shows *last* (*smods p q*) ≠*0*
  ⟨*proof*⟩

3.2 Alternative signed remainder sequences

function *smods-ext*::*real poly* ⇒ *real poly* ⇒ *real poly list* **where**
  smods-ext p q = (*if p*=*0 then* [] *else*
      (*if p mod q* ≠ *0*
        then Cons p (*smods-ext q* (−(*p mod q*)))
        else Cons p (*smods-ext q* (*pderiv q*)))
)
  ⟨*proof*⟩
termination
  ⟨*proof*⟩

lemma *smods-ext-prefix*:
  fixes *p q*::*real poly*
  defines *pp* ≡ *last* (*smods p q*)
  assumes *p*≠*0 q*≠*0*
  shows *smods-ext p q* = *smods p q* @ *tl* (*smods-ext pp* (*pderiv pp*))
  ⟨*proof*⟩

lemma *no-0-in-smods-ext*: *0*∉*set* (*smods-ext p q*)
  ⟨*proof*⟩

3.3 Sign variations on the alternative signed remainder sequences

definition *changes-itv-smods-ext*:: *real* ⇒ *real* ⇒*real poly* ⇒ *real poly* ⇒ *int* **where**
  changes-itv-smods-ext a b p q= (*let ps*= *smods-ext p q in changes-poly-at ps a*
    − *changes-poly-at ps b*)

definition *changes-gt-smods-ext*:: *real* ⇒*real poly* ⇒ *real poly* ⇒ *int* **where**
  changes-gt-smods-ext a p q= (*let ps*= *smods-ext p q in changes-poly-at ps a*
    − *changes-poly-pos-inf ps*)

definition *changes-le-smods-ext*:: *real* ⇒*real poly* ⇒ *real poly* ⇒ *int* **where**
  changes-le-smods-ext b p q= (*let ps*= *smods-ext p q in changes-poly-neg-inf ps*

$-$ *changes-poly-at ps b*)

definition *changes-R-smods-ext*:: *real poly* ⇒ *real poly* ⇒ *int* **where**
changes-R-smods-ext p q= (*let ps*= *smods-ext p q in changes-poly-neg-inf ps*
$-$ *changes-poly-pos-inf ps*)

3.4 Extension of Sturm's theorem for multiple roots

theorem *sturm-ext-interval*:
assumes $a<b$ *poly p* $a\neq 0$ *poly p* $b\neq 0$
shows *proots-count p* $\{x.\ a<x \wedge x<b\}$ = *changes-itv-smods-ext a b p* (*pderiv p*)
⟨*proof*⟩

theorem *sturm-ext-above*:
assumes *poly p* $a\neq 0$
shows *proots-count p* $\{x.\ a<x\}$ = *changes-gt-smods-ext a p* (*pderiv p*)
⟨*proof*⟩

theorem *sturm-ext-below*:
assumes *poly p* $b\neq 0$
shows *proots-count p* $\{x.\ x<b\}$ = *changes-le-smods-ext b p* (*pderiv p*)
⟨*proof*⟩

theorem *sturm-ext-R*:
assumes $p\neq 0$
shows *proots-count p UNIV* = *changes-R-smods-ext p* (*pderiv p*)
⟨*proof*⟩

end

4 Descartes Roots Test

theory *Descartes-Roots-Test* **imports** *Budan-Fourier*
begin

The Descartes roots test is a consequence of Descartes' rule of signs: through counting sign variations on coefficients of a base-transformed (i.e. Taylor shifted) polynomial, it can over-approximate the number of real roots (counting multiplicity) within an interval. Its ability is similar to the Budan-Fourier theorem, but is far more efficient in practice. Therefore, this test is widely used in modern root isolation procedures.

More information can be found in the wiki page about Vincent's theorem: https://en.wikipedia.org/wiki/Vincent%27s_theorem and Collins and Akritas's classic paper of root isolation: Collins, G.E., Akritas, A.G.: Polynomial real root isolation using Descarte's rule of signs. SYMSACC. 272–275 (1976). A more modern treatment is available from a recent implementation of isolating real roots: Kobel, A., Rouillier, F., Sagraloff, M.: Computing Real Roots of Real Polynomials ... and now For Real! Proceedings of ISSAC

'16, New York, New York, USA (2016).

lemma *bij-betw-pos-interval*:
fixes $a\ b::\mathit{real}$
assumes $a<b$
shows $\mathit{bij\text{-}betw}\ (\lambda x.\ (a+b * x)\ /\ (1+x))\ \{x.\ x>0\}\ \{x.\ a<x \wedge x<b\}$
⟨*proof*⟩

lemma *proots-sphere-pos-interval*:
fixes $a\ b::\mathit{real}$
defines $q1\equiv[:a,b:]$ **and** $q2\equiv[:1,1:]$
assumes $p\neq 0\ a<b$
shows $\mathit{proots\text{-}count}\ p\ \{x.\ a < x \wedge x < b\} = \mathit{proots\text{-}count}\ (\mathit{fcompose}\ p\ q1\ q2)\ \{x.\ 0 < x\}$
⟨*proof*⟩

definition $\mathit{descartes\text{-}roots\text{-}test}::\mathit{real} \Rightarrow \mathit{real} \Rightarrow \mathit{real\ poly} \Rightarrow \mathit{nat}$ **where**
$\mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p = \mathit{nat}\ (\mathit{changes}\ (\mathit{coeffs}\ (\mathit{fcompose}\ p\ [:a,b:]\ [:1,1:])))$

theorem *descartes-roots-test*:
fixes $p::\mathit{real\ poly}$
assumes $p\neq 0\ a<b$
shows $\mathit{proots\text{-}count}\ p\ \{x.\ a < x \wedge x <b\} \leq \mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p\ \wedge$
$\mathit{even}\ (\mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p - \mathit{proots\text{-}count}\ p\ \{x.\ a < x \wedge x < b\})$
⟨*proof*⟩

The roots test *descartes-roots-test* is exact if its result is 0 or 1.

corollary *descartes-roots-test-zero*:
fixes $p::\mathit{real\ poly}$
assumes $p\neq 0\ a<b\ \mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p = 0$
shows $\forall x.\ a<x \wedge x<b \longrightarrow \mathit{poly}\ p\ x\neq 0$
⟨*proof*⟩

corollary *descartes-roots-test-one*:
fixes $p::\mathit{real\ poly}$
assumes $p\neq 0\ a<b\ \mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p = 1$
shows $\mathit{proots\text{-}count}\ p\ \{x.\ a < x \wedge x <b\} = 1$
⟨*proof*⟩

Similar to the Budan-Fourier theorem, the Descartes roots test result is exact when all roots are real.

corollary *descartes-roots-test-real*:
fixes $p::\mathit{real\ poly}$
assumes $p\neq 0\ a<b$
assumes $\mathit{all\text{-}roots\text{-}real}\ p$
shows $\mathit{proots\text{-}count}\ p\ \{x.\ a < x \wedge x <b\} = \mathit{descartes\text{-}roots\text{-}test}\ a\ b\ p$
⟨*proof*⟩

end

5 Acknowledgements

The work was supported by the ERC Advanced Grant ALEXANDRIA (Project 742178), funded by the European Research Council and led by Professor Lawrence Paulson at the University of Cambridge, UK.

References

[1] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[2] M. Eberl. Sturm's theorem. *Archive of Formal Proofs*, Jan. 2014. http://isa-afp.org/entries/Sturm_Sequences.html, Formal proof development.

[3] M. Eberl. Descartes' rule of signs. *Archive of Formal Proofs*, Dec. 2015. http://isa-afp.org/entries/Descartes_Sign_Rule.html, Formal proof development.

[4] J. Harrison. Verifying the accuracy of polynomial approximations in HOL. In E. L. Gunter and A. Felty, editors, *Theorem Proving in Higher Order Logics: 10th International Conference, TPHOLs'97*, volume 1275 of *Lecture Notes in Computer Science*, pages 137–152, Murray Hill, NJ, 1997. Springer-Verlag.

[5] W. Li. The Sturm-Tarski Theorem. *Archive of Formal Proofs*, Sept. 2014.

[6] W. Li. Count the Number of Complex Roots. *Archive of Formal Proofs*, Oct. 2017.

[7] W. Li and L. C. Paulson. Evaluating Winding Numbers and Counting Complex Roots through Cauchy Indices in Isabelle/HOL. *CoRR*, abs/1804.03922, 2018.

[8] A. Mahboubi and C. Cohen. Formal proofs in real algebraic geometry: from ordered fields to quantifier elimination. *Logical Methods in Computer Science*, 8(1), 2012.

[9] A. Narkawicz, C. A. Muñoz, and A. Dutle. Formally-Verified Decision Procedures for Univariate Polynomial Computation Based on Sturm's and Tarski's Theorems. *Journal of Automated Reasoning*, 54(4):285–326, 2015.

[10] Q. I. Rahman and G. Schmeisser. *Analytic Theory of Polynomials*. Oxford University Press, 2002.