

Evaluate Winding Numbers through Cauchy Indices

Wenda Li

December 14, 2021

Abstract

In complex analysis, the winding number measures the number of times a path (counterclockwise) winds around a point, while the Cauchy index can approximate how the path winds. This entry provides a formalisation of the Cauchy index, which is then shown to be related to the winding number. In addition, this entry also offers a tactic that enables users to evaluate the winding number by calculating Cauchy indices. The connection between the winding number and the Cauchy index can be found in the literature [1] [2, Chapter 11].

1 Some useful lemmas in topology

theory *Missing-Topology* **imports** *HOL-Analysis.Multivariate-Analysis*
begin

1.1 Misc

lemma *open-times-image*:

fixes $S::'a::\text{real-normed-field set}$

assumes *open S* $c \neq 0$

shows *open* $((\ast) c) \text{ ' } S$

<proof>

lemma *image-linear-greaterThan*:

fixes $x::'a::\text{linordered-field}$

assumes $c \neq 0$

shows $((\lambda x. c*x+b) \text{ ' } \{x < ..\}) = (\text{if } c > 0 \text{ then } \{c*x+b < ..\} \text{ else } \{.. < c*x+b\})$

<proof>

lemma *image-linear-lessThan*:

fixes $x::'a::\text{linordered-field}$

assumes $c \neq 0$

shows $((\lambda x. c*x+b) \text{ ' } \{.. < x\}) = (\text{if } c > 0 \text{ then } \{.. < c*x+b\} \text{ else } \{c*x+b < ..\})$

<proof>

lemma *continuous-on-neq-split*:

fixes $f :: 'a::\text{linear-continuum-topology} \Rightarrow 'b::\text{linorder-topology}$

assumes $\forall x \in s. f x \neq y$ *continuous-on s f connected s*
shows $(\forall x \in s. f x > y) \vee (\forall x \in s. f x < y)$
 <proof>

lemma
fixes $f :: 'a :: \text{linorder-topology} \Rightarrow 'b :: \text{topological-space}$
assumes *continuous-on {a..b} f a < b*
shows *continuous-on-at-left:continuous (at-left b) f*
and *continuous-on-at-right:continuous (at-right a) f*
 <proof>

1.2 More about *eventually*

lemma *eventually-comp-filtermap*:
eventually (P o f) F \longleftrightarrow eventually P (filtermap f F)
 <proof>

lemma *eventually-uminus-at-top-at-bot*:
fixes $P :: 'a :: \{\text{ordered-ab-group-add, linorder}\} \Rightarrow \text{bool}$
shows *eventually (P o uminus) at-bot \longleftrightarrow eventually P at-top*
eventually (P o uminus) at-top \longleftrightarrow eventually P at-bot
 <proof>

lemma *eventually-at-infinityI*:
fixes $P :: 'a :: \text{real-normed-vector} \Rightarrow \text{bool}$
assumes $\bigwedge x. c \leq \text{norm } x \implies P x$
shows *eventually P at-infinity*
 <proof>

lemma *eventually-at-bot-linorderI*:
fixes $c :: 'a :: \text{linorder}$
assumes $\bigwedge x. x \leq c \implies P x$
shows *eventually P at-bot*
 <proof>

lemma *eventually-times-inverse-1*:
fixes $f :: 'a \Rightarrow 'b :: \{\text{field, t2-space}\}$
assumes $(f \longrightarrow c) F c \neq 0$
shows $\forall_F x \text{ in } F. \text{inverse } (f x) * f x = 1$
 <proof>

1.3 More about *filtermap*

lemma *filtermap-linear-at-within*:
assumes *bij f and cont: isCont f a and open-map: $\bigwedge S. \text{open } S \implies \text{open } (f'S)$*
shows *filtermap f (at a within S) = at (f a) within f'S*
 <proof>

lemma *filtermap-at-bot-linear-eq*:
fixes $c :: 'a :: \text{linordered-field}$

assumes $c \neq 0$
shows $\text{filtermap } (\lambda x. x * c + b) \text{ at-bot} = (\text{if } c > 0 \text{ then at-bot else at-top})$
 $\langle \text{proof} \rangle$

lemma *filtermap-linear-at-left*:
fixes $c :: 'a :: \{\text{linordered-field}, \text{linorder-topology}, \text{real-normed-field}\}$
assumes $c \neq 0$
shows $\text{filtermap } (\lambda x. c * x + b) (\text{at-left } x) = (\text{if } c > 0 \text{ then at-left } (c * x + b) \text{ else at-right } (c * x + b))$
 $\langle \text{proof} \rangle$

lemma *filtermap-linear-at-right*:
fixes $c :: 'a :: \{\text{linordered-field}, \text{linorder-topology}, \text{real-normed-field}\}$
assumes $c \neq 0$
shows $\text{filtermap } (\lambda x. c * x + b) (\text{at-right } x) = (\text{if } c > 0 \text{ then at-right } (c * x + b) \text{ else at-left } (c * x + b))$
 $\langle \text{proof} \rangle$

lemma *filtermap-at-top-linear-eq*:
fixes $c :: 'a :: \text{linordered-field}$
assumes $c \neq 0$
shows $\text{filtermap } (\lambda x. x * c + b) \text{ at-top} = (\text{if } c > 0 \text{ then at-top else at-bot})$
 $\langle \text{proof} \rangle$

lemma *filtermap-nhds-open-map*:
assumes $\text{cont}: \text{isCont } f \ a$
and $\text{open-map}: \bigwedge S. \text{open } S \implies \text{open } (f'S)$
shows $\text{filtermap } f (\text{nhds } a) = \text{nhds } (f a)$
 $\langle \text{proof} \rangle$

1.4 More about *filterlim*

lemma *filterlim-at-infinity-times*:
fixes $f :: 'a \Rightarrow 'b :: \text{real-normed-field}$
assumes $\text{filterlim } f \text{ at-infinity } F \ \text{filterlim } g \text{ at-infinity } F$
shows $\text{filterlim } (\lambda x. f x * g x) \text{ at-infinity } F$
 $\langle \text{proof} \rangle$

lemma *filterlim-at-top-at-bot[elim]*:
fixes $f :: 'a \Rightarrow 'b :: \{\text{unbounded-dense-linorder}, \text{order-topology}\}$ **and** $F :: 'a \text{ filter}$
assumes $\text{top}: \text{filterlim } f \text{ at-top } F$ **and** $\text{bot}: \text{filterlim } f \text{ at-bot } F$ **and** $F \neq \text{bot}$
shows False
 $\langle \text{proof} \rangle$

lemma *filterlim-at-top-nhds[elim]*:
fixes $f :: 'a \Rightarrow 'b :: \{\text{unbounded-dense-linorder}, \text{order-topology}\}$ **and** $F :: 'a \text{ filter}$
assumes $\text{top}: \text{filterlim } f \text{ at-top } F$ **and** $\text{tendsto}: (f \longrightarrow c) F$ **and** $F \neq \text{bot}$
shows False

<proof>

lemma *filterlim-at-bot-nhds*[*elim*]:

fixes $f::'a \Rightarrow 'b::\{\text{unbounded-dense-linorder}, \text{order-topology}\}$ **and** $F::'a \text{ filter}$
assumes $\text{top:filterlim } f \text{ at-bot } F$ **and** $\text{tendsto: } (f \longrightarrow c) F$ **and** $F \neq \text{bot}$
shows *False*

<proof>

lemma *filterlim-at-top-linear-iff*:

fixes $f::'a::\text{linordered-field} \Rightarrow 'b$
assumes $c \neq 0$
shows $(\text{LIM } x \text{ at-top. } f (x * c + b) :> F2) \longleftrightarrow (\text{if } c > 0 \text{ then } (\text{LIM } x \text{ at-top. } f x :> F2) \text{ else } (\text{LIM } x \text{ at-bot. } f x :> F2))$

<proof>

lemma *filterlim-at-bot-linear-iff*:

fixes $f::'a::\text{linordered-field} \Rightarrow 'b$
assumes $c \neq 0$
shows $(\text{LIM } x \text{ at-bot. } f (x * c + b) :> F2) \longleftrightarrow (\text{if } c > 0 \text{ then } (\text{LIM } x \text{ at-bot. } f x :> F2) \text{ else } (\text{LIM } x \text{ at-top. } f x :> F2))$

<proof>

lemma *filterlim-tendsto-add-at-top-iff*:

assumes $f: (f \longrightarrow c) F$
shows $(\text{LIM } x F. (f x + g x :: \text{real}) :> \text{at-top}) \longleftrightarrow (\text{LIM } x F. g x :> \text{at-top})$

<proof>

lemma *filterlim-tendsto-add-at-bot-iff*:

fixes $c::\text{real}$
assumes $f: (f \longrightarrow c) F$
shows $(\text{LIM } x F. f x + g x :> \text{at-bot}) \longleftrightarrow (\text{LIM } x F. g x :> \text{at-bot})$

<proof>

lemma *tendsto-inverse-0-at-infinity*:

$\text{LIM } x F. f x :> \text{at-infinity} \implies ((\lambda x. \text{inverse } (f x) :: \text{real}) \longrightarrow 0) F$

<proof>

lemma *filterlim-at-infinity-divide-iff*:

fixes $f::'a \Rightarrow 'b::\text{real-normed-field}$
assumes $(f \longrightarrow c) F$ $c \neq 0$
shows $(\text{LIM } x F. f x / g x :> \text{at-infinity}) \longleftrightarrow (\text{LIM } x F. g x :> \text{at } 0)$

<proof>

lemma *filterlim-tendsto-pos-mult-at-top-iff*:

fixes $f::'a \Rightarrow \text{real}$

assumes $(f \longrightarrow c) F$ **and** $0 < c$
shows $(LIM\ x\ F.\ (f\ x\ * \ g\ x)\ :>\ at\ top) \longleftrightarrow (LIM\ x\ F.\ g\ x\ :>\ at\ top)$
 $\langle proof \rangle$

lemma *filterlim-tendsto-pos-mult-at-bot-iff*:
fixes $c :: real$
assumes $(f \longrightarrow c) F$ $0 < c$
shows $(LIM\ x\ F.\ f\ x\ * \ g\ x\ :>\ at\ bot) \longleftrightarrow filterlim\ g\ at\ bot\ F$
 $\langle proof \rangle$

lemma *filterlim-tendsto-neg-mult-at-top-iff*:
fixes $f :: 'a \Rightarrow real$
assumes $(f \longrightarrow c) F$ **and** $c < 0$
shows $(LIM\ x\ F.\ (f\ x\ * \ g\ x)\ :>\ at\ top) \longleftrightarrow (LIM\ x\ F.\ g\ x\ :>\ at\ bot)$
 $\langle proof \rangle$

lemma *filterlim-tendsto-neg-mult-at-bot-iff*:
fixes $c :: real$
assumes $(f \longrightarrow c) F$ $0 > c$
shows $(LIM\ x\ F.\ f\ x\ * \ g\ x\ :>\ at\ bot) \longleftrightarrow filterlim\ g\ at\ top\ F$
 $\langle proof \rangle$

lemma *Lim-add*:
fixes $f\ g :: 'a \Rightarrow \{t2\text{-space}, topological\text{-monoid-add}\}$
assumes $\exists y.\ (f \longrightarrow y) F$ **and** $\exists y.\ (g \longrightarrow y) F$ **and** $F \neq bot$
shows $Lim\ F\ f\ +\ Lim\ F\ g = Lim\ F\ (\lambda x.\ f\ x\ +\ g\ x)$
 $\langle proof \rangle$

1.5 Isolate and discrete

definition (in *topological-space*) *isolate*:: $'a\ set \Rightarrow bool$ (**infixr** *isolate* 60)
where $x\ isolate\ S \longleftrightarrow (x \in S \wedge (\exists T.\ open\ T \wedge T \cap S = \{x\}))$

definition (in *topological-space*) *discrete*:: $'a\ set \Rightarrow bool$
where $discrete\ S \longleftrightarrow (\forall x \in S.\ x\ isolate\ S)$

definition (in *metric-space*) *uniform-discrete* :: $'a\ set \Rightarrow bool$ **where**
 $uniform\text{-discrete}\ S \longleftrightarrow (\exists e > 0.\ \forall x \in S.\ \forall y \in S.\ dist\ x\ y < e \longrightarrow x = y)$

lemma *uniformI1*:
assumes $e > 0 \wedge x\ y.\ [x \in S; y \in S; dist\ x\ y < e] \Longrightarrow x = y$
shows $uniform\text{-discrete}\ S$
 $\langle proof \rangle$

lemma *uniformI2*:
assumes $e > 0 \wedge x\ y.\ [x \in S; y \in S; x \neq y] \Longrightarrow dist\ x\ y \geq e$
shows $uniform\text{-discrete}\ S$
 $\langle proof \rangle$

lemma *isolate-islimpt-iff*: $(x \text{ isolate } S) \longleftrightarrow (\neg (x \text{ islimpt } S) \wedge x \in S)$
<proof>

lemma *isolate-dist-Ex-iff*:
fixes $x :: 'a :: \text{metric-space}$
shows $x \text{ isolate } S \longleftrightarrow (x \in S \wedge (\exists e > 0. \forall y \in S. \text{dist } x \ y < e \longrightarrow y = x))$
<proof>

lemma *discrete-empty[simp]*: *discrete* {}
<proof>

lemma *uniform-discrete-empty[simp]*: *uniform-discrete* {}
<proof>

lemma *isolate-insert*:
fixes $x :: 'a :: \text{t1-space}$
shows $x \text{ isolate } (\text{insert } a \ S) \longleftrightarrow x \text{ isolate } S \vee (x = a \wedge \neg (x \text{ islimpt } S))$
<proof>

lemma *uniform-discrete-imp-closed*:
uniform-discrete $S \implies \text{closed } S$
<proof>

lemma *uniform-discrete-imp-discrete*:
uniform-discrete $S \implies \text{discrete } S$
<proof>

lemma *isolate-subset*: $x \text{ isolate } S \implies T \subseteq S \implies x \in T \implies x \text{ isolate } T$
<proof>

lemma *discrete-subset[elim]*: *discrete* $S \implies T \subseteq S \implies \text{discrete } T$
<proof>

lemma *uniform-discrete-subset[elim]*: *uniform-discrete* $S \implies T \subseteq S \implies \text{uniform-discrete } T$
<proof>

lemma *continuous-on-discrete*: *discrete* $S \implies \text{continuous-on } S \ f$
<proof>

lemma *uniform-discrete-insert*:
fixes $S :: 'a :: \text{euclidean-space set}$
shows $\text{uniform-discrete } (\text{insert } a \ S) \longleftrightarrow \text{uniform-discrete } S$
<proof>

lemma *discrete-compact-finite-iff*:

fixes $S :: 'a::t1\text{-space set}$
shows $\text{discrete } S \wedge \text{compact } S \longleftrightarrow \text{finite } S$
 $\langle \text{proof} \rangle$

lemma *uniform-discrete-finite-iff*:
fixes $S :: 'a::\text{heine-borel set}$
shows $\text{uniform-discrete } S \wedge \text{bounded } S \longleftrightarrow \text{finite } S$
 $\langle \text{proof} \rangle$

lemma *uniform-discrete-image-scale*:
fixes $f :: 'a::\text{euclidean-space} \Rightarrow 'b::\text{euclidean-space}$
assumes $\text{uniform-discrete } S$ **and** $\text{dist}:\forall x\in S. \forall y\in S. \text{dist } x \ y = c * \text{dist } (f \ x) \ (f \ y)$
shows $\text{uniform-discrete } (f \ 'S)$
 $\langle \text{proof} \rangle$

end

2 Some useful lemmas in algebra

theory *Missing-Algebraic imports*
HOL-Computational-Algebra.Polynomial-Factorial
HOL-Computational-Algebra.Fundamental-Theorem-Algebra
HOL-Complex-Analysis.Complex-Analysis
Missing-Topology
Budan-Fourier.BF-Misc
begin

2.1 Misc

lemma *poly-holomorphic-on[simp]*:
 $(\text{poly } p) \text{ holomorphic-on } s$
 $\langle \text{proof} \rangle$

lemma *order-zorder*:
fixes $p::\text{complex poly}$ **and** $z::\text{complex}$
assumes $p \neq 0$
shows $\text{order } z \ p = \text{nat } (\text{zorder } (\text{poly } p) \ z)$
 $\langle \text{proof} \rangle$

lemma *pcompose-pCons-0*: $\text{pcompose } p \ [a:] = [:\text{poly } p \ a:]$
 $\langle \text{proof} \rangle$

lemma *pcompose-coeff-0*:
 $\text{coeff } (\text{pcompose } p \ q) \ 0 = \text{poly } p \ (\text{coeff } q \ 0)$
 $\langle \text{proof} \rangle$

lemma *poly-field-differentiable-at[simp]*:
poly p field-differentiable (at x within s)
 ⟨proof⟩

lemma *deriv-pderiv*:
 $deriv (poly p) = poly (pderiv p)$
 ⟨proof⟩

lemma *lead-coeff-map-poly-nz*:
assumes $f (lead-coeff p) \neq 0$ $f 0 = 0$
shows $lead-coeff (map-poly f p) = f (lead-coeff p)$
 ⟨proof⟩

lemma *filterlim-poly-at-infinity*:
fixes $p::'a::real-normed-field$ *poly*
assumes $degree p > 0$
shows *filterlim (poly p) at-infinity at-infinity*
 ⟨proof⟩

lemma *poly-divide-tendsto-aux*:
fixes $p::'a::real-normed-field$ *poly*
shows $((\lambda x. poly p x / x^{(degree p)}) \longrightarrow lead-coeff p)$ *at-infinity*
 ⟨proof⟩

lemma *filterlim-power-at-infinity*:
assumes $n \neq 0$
shows *filterlim* $(\lambda x::'a::real-normed-field. x^n)$ *at-infinity at-infinity*
 ⟨proof⟩

lemma *poly-divide-tendsto-0-at-infinity*:
fixes $p::'a::real-normed-field$ *poly*
assumes $degree p > degree q$
shows $((\lambda x. poly q x / poly p x) \longrightarrow 0)$ *at-infinity*
 ⟨proof⟩

lemma *lead-coeff-list-def*:
 $lead-coeff p = (if coeffs p = [] then 0 else last (coeffs p))$
 ⟨proof⟩

lemma *poly-linepath-comp*:
fixes $a::'a::\{real-normed-vector, comm-semiring-0, real-algebra-1\}$
shows $poly p \circ (linepath a b) = poly (p \circ_p [:a, b-a:]) \circ of-real$
 ⟨proof⟩

lemma *poly-eventually-not-zero*:
fixes $p::real$ *poly*
assumes $p \neq 0$
shows *eventually* $(\lambda x. poly p x \neq 0)$ *at-infinity*
 ⟨proof⟩

2.2 More about degree

lemma *degree-div-less*:
 fixes $x y::'a::field$ $poly$
 assumes $degree\ x \neq 0$ $degree\ y \neq 0$
 shows $degree\ (x\ div\ y) < degree\ x$
 $\langle proof \rangle$

lemma *map-poly-degree-eq*:
 assumes $f\ (lead-coeff\ p) \neq 0$
 shows $degree\ (map-poly\ f\ p) = degree\ p$
 $\langle proof \rangle$

lemma *map-poly-degree-less*:
 assumes $f\ (lead-coeff\ p) = 0$ $degree\ p \neq 0$
 shows $degree\ (map-poly\ f\ p) < degree\ p$
 $\langle proof \rangle$

lemma *map-poly-degree-leq[simp]*:
 shows $degree\ (map-poly\ f\ p) \leq degree\ p$
 $\langle proof \rangle$

2.3 roots / zeros of a univariate function

definition *roots-within*::($'a \Rightarrow 'b::zero$) $\Rightarrow 'a\ set \Rightarrow 'a\ set$ **where**
 $roots-within\ f\ s = \{x \in s. f\ x = 0\}$

abbreviation *roots*::($'a \Rightarrow 'b::zero$) $\Rightarrow 'a\ set$ **where**
 $roots\ f \equiv roots-within\ f\ UNIV$

2.4 The argument principle specialised to polynomials.

lemma *argument-principle-poly*:
 assumes $p \neq 0$ **and** *valid*: $valid-path\ g$ **and** *loop*: $pathfinish\ g = pathstart\ g$
 and *no-proots*: $path-image\ g \subseteq -\ roots\ p$
 shows $contour-integral\ g\ (\lambda x. deriv\ (poly\ p)\ x / poly\ p\ x) = 2 * of-real\ pi * i *$
 $(\sum_{x \in roots\ p. winding-number\ g\ x * of-nat\ (order\ x\ p))$
 $\langle proof \rangle$

end

3 Some useful lemmas about transcendental functions

theory *Missing-Transcendental* **imports**
 Missing-Topology
 Missing-Algebraic
begin

3.1 Misc

lemma *Im-Ln-eq-pi-half:*

$$z \neq 0 \implies (\text{Im}(\text{Ln } z) = \pi/2 \iff 0 < \text{Im}(z) \wedge \text{Re}(z) = 0)$$

$$z \neq 0 \implies (\text{Im}(\text{Ln } z) = -\pi/2 \iff \text{Im}(z) < 0 \wedge \text{Re}(z) = 0)$$

<proof>

lemma *Im-Ln-eq:*

assumes $z \neq 0$

shows $\text{Im}(\text{Ln } z) =$ (if $\text{Re } z \neq 0$ then
 if $\text{Re } z > 0$ then
 $\arctan(\text{Im } z / \text{Re } z)$
 else if $\text{Im } z \geq 0$ then
 $\arctan(\text{Im } z / \text{Re } z) + \pi$
 else
 $\arctan(\text{Im } z / \text{Re } z) - \pi$
 else
 if $\text{Im } z > 0$ then $\pi/2$ else $-\pi/2$)

<proof>

lemma *exp-Arg2pi2pi-multivalued:*

assumes $\exp(i * \text{of-real } x) = z$

shows $\exists k::\text{int. } x = \text{Arg}2\pi z + 2*k*\pi$

<proof>

lemma *cos-eq-neg-periodic-intro:*

assumes $x - y = 2*(\text{of-int } k)*\pi + \pi \vee x + y = 2*(\text{of-int } k)*\pi + \pi$

shows $\cos x = -\cos y$ *<proof>*

lemma *cos-eq-periodic-intro:*

assumes $x - y = 2*(\text{of-int } k)*\pi \vee x + y = 2*(\text{of-int } k)*\pi$

shows $\cos x = \cos y$ *<proof>*

lemma *sin-tan-half:* $\sin(2*x) = 2 * \tan x / (1 + (\tan x)^2)$

<proof>

lemma *cos-tan-half:* $\cos x \neq 0 \implies \cos(2*x) = (1 - (\tan x)^2) / (1 + (\tan x)^2)$

<proof>

lemma *tan-eq-arctan-Ex:*

shows $\tan x = y \iff (\exists k::\text{int. } x = \arctan y + k*\pi \vee (x = \pi/2 + k*\pi \wedge y=0))$

<proof>

lemma *arccos-unique:*

assumes $0 \leq x$

and $x \leq \pi$

and $\cos x = y$

shows $\arccos y = x$

<proof>

lemma *cos-eq-arccos-Ex*:

$\cos x = y \iff -1 \leq y \wedge y \leq 1 \wedge (\exists k::int. x = \arccos y + 2*k*\pi \vee x = -\arccos y + 2*k*\pi)$
<proof>

lemma *uniform-discrete-tan-eq*:

uniform-discrete { $x::real. \tan x = y$ }
<proof>

lemma *get-norm-value*:

fixes $a::'a::\{\text{floor-ceiling}\}$
assumes $pp > 0$
obtains $k::int$ **and** $a1$ **where** $a = (\text{of-int } k) * pp + a1$ $a0 \leq a1$ $a1 < a0 + pp$
<proof>

lemma *filtermap-tan-at-right*:

fixes $a::real$
assumes $\cos a \neq 0$
shows *filtermap* \tan (*at-right* a) = *at-right* ($\tan a$)
<proof>

lemma *filtermap-tan-at-left*:

fixes $a::real$
assumes $\cos a \neq 0$
shows *filtermap* \tan (*at-left* a) = *at-left* ($\tan a$)
<proof>

lemma *cos-zero-iff-int2*:

fixes $x::real$
shows $\cos x = 0 \iff (\exists n::int. x = n * \pi + \pi/2)$
<proof>

lemma *filtermap-tan-at-right-inf*:

fixes $a::real$
assumes $\cos a = 0$
shows *filtermap* \tan (*at-right* a) = *at-bot*
<proof>

lemma *filtermap-tan-at-left-inf*:

fixes $a::real$
assumes $\cos a = 0$
shows *filtermap* \tan (*at-left* a) = *at-top*
<proof>

3.2 Periodic set

definition *periodic-set*:: *real set* \Rightarrow *real* \Rightarrow *bool* **where**

periodic-set $S \delta \longleftrightarrow (\exists B. \text{finite } B \wedge (\forall x \in S. \exists b \in B. \exists k :: \text{int}. x = b + k * \delta))$

lemma *periodic-set-multiple*:

assumes $k \neq 0$

shows *periodic-set* $S \delta \longleftrightarrow \text{periodic-set } S \text{ (of-int } k * \delta)$

<proof>

lemma *periodic-set-empty[simp]*: *periodic-set* $\{\} \delta$

<proof>

lemma *periodic-set-finite*:

assumes *finite* S

shows *periodic-set* $S \delta$

<proof>

lemma *periodic-set-subset[elim]*:

assumes *periodic-set* $S \delta$ $T \subseteq S$

shows *periodic-set* $T \delta$

<proof>

lemma *periodic-set-union*:

assumes *periodic-set* $S \delta$ *periodic-set* $T \delta$

shows *periodic-set* $(S \cup T) \delta$

<proof>

lemma *periodic-imp-uniform-discrete*:

assumes *periodic-set* $S \delta$

shows *uniform-discrete* S

<proof>

lemma *periodic-set-tan-linear*:

assumes $a \neq 0$ $c \neq 0$

shows *periodic-set* $(\text{roots } (\lambda x. a * \tan(x/c) + b)) (c * \pi)$

<proof>

lemma *periodic-set-cos-linear*:

assumes $a \neq 0$ $c \neq 0$

shows *periodic-set* $(\text{roots } (\lambda x. a * \cos(x/c) + b)) (2 * c * \pi)$

<proof>

lemma *periodic-set-tan-poly*:

assumes $p \neq 0$ $c \neq 0$

shows *periodic-set* $(\text{roots } (\lambda x. \text{poly } p (\tan(x/c)))) (c * \pi)$

<proof>

lemma *periodic-set-sin-cos-linear*:

fixes $a b c :: \text{real}$

assumes $a \neq 0 \vee b \neq 0 \vee c \neq 0$

shows *periodic-set* $(\text{roots } (\lambda x. a * \cos x + b * \sin x + c)) (4 * \pi)$

<proof>

end

4 Some useful lemmas in analysis

theory *Missing-Analysis*
imports *HOL-Complex-Analysis.Complex-Analysis*
begin

4.1 More about paths

lemma *pathfinish-offset[simp]*:
 $\text{pathfinish } (\lambda t. g t - z) = \text{pathfinish } g - z$
<proof>

lemma *pathstart-offset[simp]*:
 $\text{pathstart } (\lambda t. g t - z) = \text{pathstart } g - z$
<proof>

lemma *pathimage-offset[simp]*:
fixes $g :: - \Rightarrow 'b::\text{topological-group-add}$
shows $p \in \text{path-image } (\lambda t. g t - z) \longleftrightarrow p+z \in \text{path-image } g$
<proof>

lemma *path-offset[simp]*:
fixes $g :: - \Rightarrow 'b::\text{topological-group-add}$
shows $\text{path } (\lambda t. g t - z) \longleftrightarrow \text{path } g$
<proof>

lemma *not-on-circlepathI*:
assumes $\text{cmod } (z-z0) \neq |r|$
shows $z \notin \text{path-image } (\text{part-circlepath } z0 r st tt)$
<proof>

lemma *circlepath-inj-on*:
assumes $r > 0$
shows $\text{inj-on } (\text{circlepath } z r) \{0..<1\}$
<proof>

4.2 More lemmas related to *winding-number*

lemma *winding-number-comp*:
assumes $\text{open } s \text{ holomorphic-on } s \text{ path-image } \gamma \subseteq s$
 $\text{valid-path } \gamma \text{ } z \notin \text{path-image } (f \circ \gamma)$
shows $\text{winding-number } (f \circ \gamma) z = 1/(2*\text{pi}*i)* \text{contour-integral } \gamma (\lambda w. \text{deriv } f w / (f w - z))$
<proof>

lemma *winding-number-uminus-comp*:
assumes *valid-path* $\gamma - z \notin \text{path-image } \gamma$
shows *winding-number* $(\text{uminus} \circ \gamma) z = \text{winding-number } \gamma (-z)$
 $\langle \text{proof} \rangle$

lemma *winding-number-comp-linear*:
assumes $c \neq 0$ *valid-path* γ **and** *not-image*: $(z-b)/c \notin \text{path-image } \gamma$
shows *winding-number* $((\lambda x. c*x+b) \circ \gamma) z = \text{winding-number } \gamma ((z-b)/c)$ (**is**
 $?L = ?R$)
 $\langle \text{proof} \rangle$

end

5 Cauchy's index theorem

theory *Cauchy-Index-Theorem* **imports**
HOL-Complex-Analysis.Complex-Analysis
Sturm-Tarski.Sturm-Tarski
HOL-Computational-Algebra.Fundamental-Theorem-Algebra
Missing-Transcendental
Missing-Algebraic
Missing-Analysis
begin

This theory formalises Cauchy indices on the complex plane and relate them to winding numbers

5.1 Misc

lemma *atMostAtLeast-subset-convex*:
fixes $C :: \text{real set}$
assumes *convex* C
and $x \in C$ $y \in C$
shows $\{x .. y\} \subseteq C$
 $\langle \text{proof} \rangle$

lemma *arg-elim*:
 $f x \implies x = y \implies f y$
 $\langle \text{proof} \rangle$

lemma *arg-elim2*:
 $f x1 x2 \implies x1 = y1 \implies x2 = y2 \implies f y1 y2$
 $\langle \text{proof} \rangle$

lemma *arg-elim3*:
 $[f x1 x2 x3; x1 = y1; x2 = y2; x3 = y3] \implies f y1 y2 y3$
 $\langle \text{proof} \rangle$

lemma *IVT-strict*:

fixes $f :: 'a::linear-continuum-topology \Rightarrow 'b::linorder-topology$
assumes $(f a > y \wedge y > f b) \vee (f a < y \wedge y < f b) \ a < b \ \text{continuous-on } \{a .. b\} \ f$
shows $\exists x. a < x \wedge x < b \wedge f x = y$
 $\langle \text{proof} \rangle$

lemma (*in dense-linorder*) *atLeastAtMost-subseteq-greaterThanLessThan-iff*:
 $\{a .. b\} \subseteq \{c <..< d\} \iff (a \leq b \longrightarrow c < a \wedge b < d)$
 $\langle \text{proof} \rangle$

lemma *Re-winding-number-half-right*:
assumes $\forall p \in \text{path-image } \gamma. \ \text{Re } p \geq \text{Re } z \ \text{and} \ \text{valid-path } \gamma \ \text{and} \ z \notin \text{path-image } \gamma$
shows $\text{Re}(\text{winding-number } \gamma \ z) = (\text{Im } (\text{Ln } (\text{pathfinish } \gamma - z)) - \text{Im } (\text{Ln } (\text{pathstart } \gamma - z)))/(2*\pi)$
 $\langle \text{proof} \rangle$

lemma *Re-winding-number-half-upper*:
assumes $\text{pimage}:\forall p \in \text{path-image } \gamma. \ \text{Im } p \geq \text{Im } z \ \text{and} \ \text{valid-path } \gamma \ \text{and} \ z \notin \text{path-image } \gamma$
shows $\text{Re}(\text{winding-number } \gamma \ z) = (\text{Im } (\text{Ln } (i*z - i*\text{pathfinish } \gamma)) - \text{Im } (\text{Ln } (i*z - i*\text{pathstart } \gamma)))/(2*\pi)$
 $\langle \text{proof} \rangle$

lemma *Re-winding-number-half-lower*:
assumes $\text{pimage}:\forall p \in \text{path-image } \gamma. \ \text{Im } p \leq \text{Im } z \ \text{and} \ \text{valid-path } \gamma \ \text{and} \ z \notin \text{path-image } \gamma$
shows $\text{Re}(\text{winding-number } \gamma \ z) = (\text{Im } (\text{Ln } (i*\text{pathfinish } \gamma - i*z)) - \text{Im } (\text{Ln } (i*\text{pathstart } \gamma - i*z)))/(2*\pi)$
 $\langle \text{proof} \rangle$

lemma *Re-winding-number-half-left*:
assumes $\text{neg-img}:\forall p \in \text{path-image } \gamma. \ \text{Re } p \leq \text{Re } z \ \text{and} \ \text{valid-path } \gamma \ \text{and} \ z \notin \text{path-image } \gamma$
shows $\text{Re}(\text{winding-number } \gamma \ z) = (\text{Im } (\text{Ln } (z - \text{pathfinish } \gamma)) - \text{Im } (\text{Ln } (z - \text{pathstart } \gamma)))/(2*\pi)$
 $\langle \text{proof} \rangle$

lemma *continuous-on-open-Collect-neq*:
fixes $f \ g :: 'a::topological-space \Rightarrow 'b::t2-space$
assumes $f: \text{continuous-on } S \ f \ \text{and} \ g: \text{continuous-on } S \ g \ \text{and} \ \text{open } S$
shows $\text{open } \{x \in S. f x \neq g x\}$
 $\langle \text{proof} \rangle$

5.2 Sign at a filter

definition $\text{has-sgnx}::(\text{real} \Rightarrow \text{real}) \Rightarrow \text{real} \Rightarrow \text{real filter} \Rightarrow \text{bool}$
(infixr has'-sgnx 55) where
 $(f \ \text{has-sgnx } c) \ F = (\text{eventually } (\lambda x. \ \text{sgn}(f x) = c) \ F)$

definition *sgnx-able* (**infixr** *sgnx'-able 55*) **where**
 $(f \text{ sgnx-able } F) = (\exists c. (f \text{ has-sgnx } c) F)$

definition *sgnx* **where**
 $\text{sgnx } f F = (\text{SOME } c. (f \text{ has-sgnx } c) F)$

lemma *has-sgnx-eq-rhs*: $(f \text{ has-sgnx } x) F \implies x = y \implies (f \text{ has-sgnx } y) F$
 $\langle \text{proof} \rangle$

named-theorems *sgnx-intros* *introduction rules for has-sgnx*
 $\langle \text{ML} \rangle$

lemma *sgnx-able-sgnx*: $f \text{ sgnx-able } F \implies (f \text{ has-sgnx } (\text{sgnx } f F)) F$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-imp-sgnx-able[elim]*:
 $(f \text{ has-sgnx } c) F \implies f \text{ sgnx-able } F$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-unique*:
assumes $F \neq \text{bot } (f \text{ has-sgnx } c1) F (f \text{ has-sgnx } c2) F$
shows $c1 = c2$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-imp-sgnx[elim]*:
 $(f \text{ has-sgnx } c) F \implies F \neq \text{bot} \implies \text{sgnx } f F = c$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-const[simp,sgnx-intros]*:
 $((\lambda \cdot. c) \text{ has-sgnx } \text{sgn } c) F$
 $\langle \text{proof} \rangle$

lemma *finite-sgnx-at-left-at-right*:
assumes $\text{finite } \{t. f t = 0 \wedge a < t \wedge t < b\}$ *continuous-on* $(\{a < \cdot < b\} - s)$ f *finite* s
and $x : x \in \{a < \cdot < b\}$
shows $f \text{ sgnx-able } (\text{at-left } x) \text{ sgnx } f (\text{at-left } x) \neq 0$
 $f \text{ sgnx-able } (\text{at-right } x) \text{ sgnx } f (\text{at-right } x) \neq 0$
 $\langle \text{proof} \rangle$

lemma *sgnx-able-poly[simp]*:
 $(\text{poly } p) \text{ sgnx-able } (\text{at-right } a)$
 $(\text{poly } p) \text{ sgnx-able } (\text{at-left } a)$
 $(\text{poly } p) \text{ sgnx-able at-top}$
 $(\text{poly } p) \text{ sgnx-able at-bot}$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-identity[intro,sgnx-intros]*:
shows $x \geq 0 \implies ((\lambda x. x) \text{ has-sgnx } 1) (\text{at-right } x)$
 $x \leq 0 \implies ((\lambda x. x) \text{ has-sgnx } -1) (\text{at-left } x)$

$\langle proof \rangle$

lemma *has-sgnx-divide*[*sgnx-intros*]:

assumes $(f \text{ has-sgnx } c1) F (g \text{ has-sgnx } c2) F$

shows $((\lambda x. f x / g x) \text{ has-sgnx } c1 / c2) F$

$\langle proof \rangle$

lemma *sgnx-able-divide*[*sgnx-intros*]:

assumes $f \text{ sgnx-able } F g \text{ sgnx-able } F$

shows $(\lambda x. f x / g x) \text{ sgnx-able } F$

$\langle proof \rangle$

lemma *sgnx-divide*:

assumes $F \neq \text{bot } f \text{ sgnx-able } F g \text{ sgnx-able } F$

shows $\text{sgnx } (\lambda x. f x / g x) F = \text{sgnx } f F / \text{sgnx } g F$

$\langle proof \rangle$

lemma *has-sgnx-times*[*sgnx-intros*]:

assumes $(f \text{ has-sgnx } c1) F (g \text{ has-sgnx } c2) F$

shows $((\lambda x. f x * g x) \text{ has-sgnx } c1 * c2) F$

$\langle proof \rangle$

lemma *sgnx-able-times*[*sgnx-intros*]:

assumes $f \text{ sgnx-able } F g \text{ sgnx-able } F$

shows $(\lambda x. f x * g x) \text{ sgnx-able } F$

$\langle proof \rangle$

lemma *sgnx-times*:

assumes $F \neq \text{bot } f \text{ sgnx-able } F g \text{ sgnx-able } F$

shows $\text{sgnx } (\lambda x. f x * g x) F = \text{sgnx } f F * \text{sgnx } g F$

$\langle proof \rangle$

lemma *tendsto-nonzero-has-sgnx*:

assumes $(f \longrightarrow c) F c \neq 0$

shows $(f \text{ has-sgnx } \text{sgn } c) F$

$\langle proof \rangle$

lemma *tendsto-nonzero-sgnx*:

assumes $(f \longrightarrow c) F F \neq \text{bot } c \neq 0$

shows $\text{sgnx } f F = \text{sgn } c$

$\langle proof \rangle$

lemma *filterlim-divide-at-bot-at-top-iff*:

assumes $(f \longrightarrow c) F c \neq 0$

shows

$(LIM x F. f x / g x \text{ :> } \text{at-bot}) \longleftrightarrow (g \longrightarrow 0) F$

$\wedge ((\lambda x. g x) \text{ has-sgnx } - \text{sgn } c) F$

$(LIM x F. f x / g x \text{ :> } \text{at-top}) \longleftrightarrow (g \longrightarrow 0) F$

$\wedge ((\lambda x. g x) \text{ has-sgnx sgn } c) F$
 $\langle \text{proof} \rangle$

lemma *poly-sgnx-left-right*:
fixes $c a::\text{real}$ **and** $p::\text{real poly}$
assumes $p \neq 0$
shows $\text{sgnx } (\text{poly } p) (\text{at-left } a) = (\text{if even } (\text{order } a \ p)$
 $\text{then } \text{sgnx } (\text{poly } p) (\text{at-right } a)$
 $\text{else } -\text{sgnx } (\text{poly } p) (\text{at-right } a))$
 $\langle \text{proof} \rangle$

lemma *poly-has-sgnx-left-right*:
fixes $c a::\text{real}$ **and** $p::\text{real poly}$
assumes $p \neq 0$
shows $(\text{poly } p \text{ has-sgnx } c) (\text{at-left } a) \longleftrightarrow (\text{if even } (\text{order } a \ p)$
 $\text{then } (\text{poly } p \text{ has-sgnx } c) (\text{at-right } a)$
 $\text{else } (\text{poly } p \text{ has-sgnx } -c) (\text{at-right } a))$
 $\langle \text{proof} \rangle$

lemma *sign-r-pos-sgnx-iff*:
 $\text{sign-r-pos } p \ a \longleftrightarrow \text{sgnx } (\text{poly } p) (\text{at-right } a) > 0$
 $\langle \text{proof} \rangle$

lemma *sgnx-values*:
assumes $f \text{ sgnx-able } F \ F \neq \text{bot}$
shows $\text{sgnx } f \ F = -1 \vee \text{sgnx } f \ F = 0 \vee \text{sgnx } f \ F = 1$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-poly-at-top*:
 $(\text{poly } p \text{ has-sgnx } \text{sgn-pos-inf } p) \text{ at-top}$
 $\langle \text{proof} \rangle$

lemma *has-sgnx-poly-at-bot*:
 $(\text{poly } p \text{ has-sgnx } \text{sgn-neg-inf } p) \text{ at-bot}$
 $\langle \text{proof} \rangle$

lemma *sgnx-poly-at-top*:
 $\text{sgnx } (\text{poly } p) \text{ at-top} = \text{sgn-pos-inf } p$
 $\langle \text{proof} \rangle$

lemma *sgnx-poly-at-bot*:
 $\text{sgnx } (\text{poly } p) \text{ at-bot} = \text{sgn-neg-inf } p$
 $\langle \text{proof} \rangle$

lemma *poly-has-sgnx-values*:
assumes $p \neq 0$
shows

$(\text{poly } p \text{ has-sgnx } 1) (\text{at-left } a) \vee (\text{poly } p \text{ has-sgnx } -1) (\text{at-left } a)$
 $(\text{poly } p \text{ has-sgnx } 1) (\text{at-right } a) \vee (\text{poly } p \text{ has-sgnx } -1) (\text{at-right } a)$
 $(\text{poly } p \text{ has-sgnx } 1) \text{ at-top} \vee (\text{poly } p \text{ has-sgnx } -1) \text{ at-top}$
 $(\text{poly } p \text{ has-sgnx } 1) \text{ at-bot} \vee (\text{poly } p \text{ has-sgnx } -1) \text{ at-bot}$
 ⟨proof⟩

lemma *poly-sgnx-values*:

assumes $p \neq 0$

shows $\text{sgnx } (\text{poly } p) (\text{at-left } a) = 1 \vee \text{sgnx } (\text{poly } p) (\text{at-left } a) = -1$

$\text{sgnx } (\text{poly } p) (\text{at-right } a) = 1 \vee \text{sgnx } (\text{poly } p) (\text{at-right } a) = -1$

⟨proof⟩

lemma *has-sgnx-inverse*: $(f \text{ has-sgnx } c) F \longleftrightarrow ((\text{inverse } o f) \text{ has-sgnx } (\text{inverse } c)) F$

⟨proof⟩

lemma *has-sgnx-derivative-at-left*:

assumes $g \text{-deriv}:(g \text{ has-field-derivative } c) (\text{at } x)$ **and** $g \text{ at } x = 0$ **and** $c \neq 0$

shows $(g \text{ has-sgnx } - \text{sgn } c) (\text{at-left } x)$

⟨proof⟩

lemma *has-sgnx-derivative-at-right*:

assumes $g \text{-deriv}:(g \text{ has-field-derivative } c) (\text{at } x)$ **and** $g \text{ at } x = 0$ **and** $c \neq 0$

shows $(g \text{ has-sgnx } \text{sgn } c) (\text{at-right } x)$

⟨proof⟩

lemma *has-sgnx-split*:

$(f \text{ has-sgnx } c) (\text{at } x) \longleftrightarrow (f \text{ has-sgnx } c) (\text{at-left } x) \wedge (f \text{ has-sgnx } c) (\text{at-right } x)$

⟨proof⟩

lemma *sgnx-at-top-IVT*:

assumes $\text{sgnx } (\text{poly } p) (\text{at-right } a) \neq \text{sgnx } (\text{poly } p) \text{ at-top}$

shows $\exists x > a. \text{poly } p \text{ at } x = 0$

⟨proof⟩

lemma *sgnx-at-left-at-right-IVT*:

assumes $\text{sgnx } (\text{poly } p) (\text{at-right } a) \neq \text{sgnx } (\text{poly } p) (\text{at-left } b)$ $a < b$

shows $\exists x. a < x \wedge x < b \wedge \text{poly } p \text{ at } x = 0$

⟨proof⟩

lemma *sgnx-at-bot-IVT*:

assumes $\text{sgnx } (\text{poly } p) (\text{at-left } a) \neq \text{sgnx } (\text{poly } p) \text{ at-bot}$

shows $\exists x < a. \text{poly } p \text{ at } x = 0$

⟨proof⟩

lemma *sgnx-poly-nz*:

assumes $\text{poly } p \text{ at } x \neq 0$

shows $\text{sgnx } (\text{poly } p) (\text{at-left } x) = \text{sgn } (\text{poly } p \text{ at } x)$

$\text{sgn } x \text{ (poly } p \text{) (at-right } x \text{) = sgn (poly } p \text{ } x \text{)}$
 ⟨proof⟩

5.3 Finite predicate segments over an interval

inductive *finite-Psegments*::(real \Rightarrow bool) \Rightarrow real \Rightarrow real \Rightarrow bool for *P* where

emptyI: $a \geq b \implies \text{finite-Psegments } P \ a \ b$

insertI-1: $\llbracket s \in \{a..<b\}; s = a \vee P \ s; \forall t \in \{s<..$

$\implies \text{finite-Psegments } P \ a \ b$

insertI-2: $\llbracket s \in \{a..<b\}; s = a \vee P \ s; (\forall t \in \{s<..$

$\implies \text{finite-Psegments } P \ a \ b$

lemma *finite-Psegments-pos-linear*:

assumes *finite-Psegments* *P* ($b * lb + c$) ($b * ub + c$) **and** $b > 0$

shows *finite-Psegments* (*P* o ($\lambda t. b * t + c$)) *lb ub*

⟨proof⟩

lemma *finite-Psegments-congE*:

assumes *finite-Psegments* *Q lb ub*

$\bigwedge t. \llbracket lb < t; t < ub \rrbracket \implies Q \ t \longleftrightarrow P \ t$

shows *finite-Psegments* *P lb ub* ⟨proof⟩

lemma *finite-Psegments-constI*:

assumes $\bigwedge t. \llbracket a < t; t < b \rrbracket \implies P \ t = c$

shows *finite-Psegments* *P a b*

⟨proof⟩

context

begin

private lemma *finite-Psegments-less-eq1*:

assumes *finite-Psegments* *P a c b* $b \leq c$

shows *finite-Psegments* *P a b* ⟨proof⟩ **lemma** *finite-Psegments-less-eq2*:

assumes *finite-Psegments* *P a c a* $a \leq b$

shows *finite-Psegments* *P b c* ⟨proof⟩

lemma *finite-Psegments-included*:

assumes *finite-Psegments* *P a d a* $a \leq b \ c \leq d$

shows *finite-Psegments* *P b c*

⟨proof⟩

end

lemma *finite-Psegments-combine*:

assumes *finite-Psegments* *P a b* *finite-Psegments* *P b c* $b \in \{a..c\}$ *closed* ($\{x. P \ x\} \cap \{a..c\}$)

shows *finite-Psegments* *P a c* ⟨proof⟩

5.4 Finite segment intersection of a path with the imaginary axis

definition *finite-ReZ-segments*:(*real* \Rightarrow *complex*) \Rightarrow *complex* \Rightarrow *bool* where
finite-ReZ-segments $g\ z = \text{finite-Psegments } (\lambda t. \text{Re } (g\ t - z) = 0)\ 0\ 1$

lemma *finite-ReZ-segments-joinpaths*:

assumes $g1:\text{finite-ReZ-segments } g1\ z$ **and** $g2:\text{finite-ReZ-segments } g2\ z$ **and**
 $\text{path } g1\ \text{path } g2\ \text{pathfinish } g1 = \text{pathstart } g2$

shows *finite-ReZ-segments* $(g1+++g2)\ z$

$\langle \text{proof} \rangle$

lemma *finite-ReZ-segments-congE*:

assumes *finite-ReZ-segments* $p1\ z1$

$\bigwedge t. \llbracket 0 < t; t < 1 \rrbracket \implies \text{Re}(p1\ t - z1) = \text{Re}(p2\ t - z2)$

shows *finite-ReZ-segments* $p2\ z2$

$\langle \text{proof} \rangle$

lemma *finite-ReZ-segments-constI*:

assumes $\forall t. 0 < t \wedge t < 1 \longrightarrow g\ t = c$

shows *finite-ReZ-segments* $g\ z$

$\langle \text{proof} \rangle$

lemma *finite-ReZ-segment-cases* [*consumes 1, case-names subEq subNEq, cases pred:finite-ReZ-segments*]:

assumes *finite-ReZ-segments* $g\ z$

and *subEq*: $(\bigwedge s. \llbracket s \in \{0..<1\}; s=0 \vee \text{Re } (g\ s) = \text{Re } z;$

$\forall t \in \{s < .. < 1\}. \text{Re } (g\ t) = \text{Re } z; \text{finite-ReZ-segments } (\text{subpath } 0\ s\ g)\ z \rrbracket \implies$

P)

and *subNEq*: $(\bigwedge s. \llbracket s \in \{0..<1\}; s=0 \vee \text{Re } (g\ s) = \text{Re } z;$

$\forall t \in \{s < .. < 1\}. \text{Re } (g\ t) \neq \text{Re } z; \text{finite-ReZ-segments } (\text{subpath } 0\ s\ g)\ z \rrbracket \implies$

P)

shows *P*

$\langle \text{proof} \rangle$

lemma *finite-ReZ-segments-induct* [*case-names sub0 subEq subNEq, induct pred:finite-ReZ-segments*]:

assumes *finite-ReZ-segments* $g\ z$

assumes *sub0*: $\bigwedge g\ z. (P\ (\text{subpath } 0\ 0\ g)\ z)$

and *subEq*: $(\bigwedge s\ g\ z. \llbracket s \in \{0..<1\}; s=0 \vee \text{Re } (g\ s) = \text{Re } z;$

$\forall t \in \{s < .. < 1\}. \text{Re } (g\ t) = \text{Re } z; \text{finite-ReZ-segments } (\text{subpath } 0\ s\ g)\ z;$

$P\ (\text{subpath } 0\ s\ g)\ z \rrbracket \implies P\ g\ z)$

and *subNEq*: $(\bigwedge s\ g\ z. \llbracket s \in \{0..<1\}; s=0 \vee \text{Re } (g\ s) = \text{Re } z;$

$\forall t \in \{s < .. < 1\}. \text{Re } (g\ t) \neq \text{Re } z; \text{finite-ReZ-segments } (\text{subpath } 0\ s\ g)\ z;$

$P\ (\text{subpath } 0\ s\ g)\ z \rrbracket \implies P\ g\ z)$

shows $P\ g\ z$

$\langle \text{proof} \rangle$

lemma *finite-ReZ-segments-shiftpah*:

assumes *finite-ReZ-segments* $g\ z\ s \in \{0..1\}$ **path** g **and** *loop*: $\text{pathfinish } g = \text{pathstart } g$

shows *finite-ReZ-segments* $(\text{shiftpath } s\ g)\ z$

<proof>

lemma *finite-imp-finite-ReZ-segments:*

assumes *finite* $\{t. \text{Re } (g t - z) = 0 \wedge 0 \leq t \wedge t \leq 1\}$

shows *finite-ReZ-segments* $g z$

<proof>

lemma *finite-ReZ-segments-poly-linepath:*

shows *finite-ReZ-segments* $(\text{poly } p \text{ o linepath } a b) z$

<proof>

lemma *part-circlepath-half-finite-inter:*

assumes $st \neq tt \ r \neq 0 \ c \neq 0$

shows *finite* $\{t. \text{part-circlepath } z0 \ r \ st \ tt \ t \cdot c = d \wedge 0 \leq t \wedge t \leq 1\}$ (**is** *finite ?T*)

<proof>

lemma *linepath-half-finite-inter:*

assumes $a \cdot c \neq d \vee b \cdot c \neq d$

shows *finite* $\{t. \text{linepath } a b \ t \cdot c = d \wedge 0 \leq t \wedge t \leq 1\}$ (**is** *finite ?S*)

<proof>

lemma *finite-half-joinpaths-inter:*

assumes *finite* $\{t. l1 \ t \cdot c = d \wedge 0 \leq t \wedge t \leq 1\}$ *finite* $\{t. l2 \ t \cdot c = d \wedge 0 \leq t \wedge t \leq 1\}$

shows *finite* $\{t. (l1 +++ l2) \ t \cdot c = d \wedge 0 \leq t \wedge t \leq 1\}$

<proof>

lemma *finite-ReZ-segments-linepath:*

finite-ReZ-segments $(\text{linepath } a b) z$

<proof>

lemma *finite-ReZ-segments-part-circlepath:*

finite-ReZ-segments $(\text{part-circlepath } z0 \ r \ st \ tt) z$

<proof>

lemma *finite-ReZ-segments-poly-of-real:*

shows *finite-ReZ-segments* $(\text{poly } p \text{ o of-real}) z$

<proof>

lemma *finite-ReZ-segments-subpath:*

assumes *finite-ReZ-segments* $g z$

$0 \leq u \ u \leq v \ v \leq 1$

shows *finite-ReZ-segments* $(\text{subpath } u \ v \ g) z$

<proof>

5.5 jump and jumpF

definition *jump::(real \Rightarrow real) \Rightarrow real \Rightarrow int* **where**

jump $f \ a = (\text{if}$

$(LIM\ x\ (at\text{-}left\ a).\ f\ x\ :\>\ at\text{-}bot) \wedge (LIM\ x\ (at\text{-}right\ a).\ f\ x\ :\>\ at\text{-}top)$
then 1 else if
 $(LIM\ x\ (at\text{-}left\ a).\ f\ x\ :\>\ at\text{-}top) \wedge (LIM\ x\ (at\text{-}right\ a).\ f\ x\ :\>\ at\text{-}bot)$
then -1 else 0)

definition $jumpF::(real \Rightarrow real) \Rightarrow real\ filter \Rightarrow real$ **where**

$jumpF\ f\ F \equiv (if\ filterlim\ f\ at\text{-}top\ F\ then\ 1/2\ else$
 $if\ filterlim\ f\ at\text{-}bot\ F\ then\ -1/2\ else\ (0::real))$

lemma $jumpF\text{-}const[simp]$:

assumes $F \neq bot$
shows $jumpF\ (\lambda\cdot.\ c)\ F = 0$

$\langle proof \rangle$

lemma $jumpF\text{-}not\text{-}infinity$:

assumes $continuous\ F\ g\ F \neq bot$
shows $jumpF\ g\ F = 0$

$\langle proof \rangle$

lemma $jumpF\text{-}linear\text{-}comp$:

assumes $c \neq 0$

shows

$jumpF\ (f\ o\ (\lambda x.\ c*x+b))\ (at\text{-}left\ x) =$
 $(if\ c>0\ then\ jumpF\ f\ (at\text{-}left\ (c*x+b))\ else\ jumpF\ f\ (at\text{-}right\ (c*x+b)))$
(is ?case1)
 $jumpF\ (f\ o\ (\lambda x.\ c*x+b))\ (at\text{-}right\ x) =$
 $(if\ c>0\ then\ jumpF\ f\ (at\text{-}right\ (c*x+b))\ else\ jumpF\ f\ (at\text{-}left\ (c*x+b)))$
(is ?case2)

$\langle proof \rangle$

lemma $jump\text{-}const[simp]:jump\ (\lambda\cdot.\ c)\ a = 0$

$\langle proof \rangle$

lemma $jump\text{-}not\text{-}infinity$:

$isCont\ f\ a \implies jump\ f\ a = 0$

$\langle proof \rangle$

lemma $jump\text{-}jump\text{-}poly\text{-}aux$:

assumes $p \neq 0\ coprime\ p\ q$

shows $jump\ (\lambda x.\ poly\ q\ x / poly\ p\ x)\ a = jump\text{-}poly\ q\ p\ a$

$\langle proof \rangle$

lemma $jump\text{-}jumpF$:

assumes $cont:isCont\ (inverse\ o\ f)\ a$ **and**

$sgnxl:(f\ has\text{-}sgnx\ l)\ (at\text{-}left\ a)$ **and** $sgnxr:(f\ has\text{-}sgnx\ r)\ (at\text{-}right\ a)$ **and**
 $l \neq 0\ r \neq 0$

shows $jump\ f\ a = jumpF\ f\ (at\text{-}right\ a) - jumpF\ f\ (at\text{-}left\ a)$

$\langle proof \rangle$

lemma *jump-linear-comp*:

assumes $c \neq 0$

shows $\text{jump } (f \circ (\lambda x. c*x+b)) \ x = (\text{if } c > 0 \text{ then } \text{jump } f \ (c*x+b) \ \text{else } -\text{jump } f \ (c*x+b))$

<proof>

lemma *jump-divide-derivative*:

assumes $\text{isCont } f \ x \ g \ x = 0 \ f \ x \neq 0$

and $g\text{-deriv}:(g \ \text{has-field-derivative } c) \ (\text{at } x) \ \text{and } c \neq 0$

shows $\text{jump } (\lambda t. f \ t/g \ t) \ x = (\text{if } \text{sgn } c = \text{sgn } (f \ x) \ \text{then } 1 \ \text{else } -1)$

<proof>

lemma *jump-jump-poly*: $\text{jump } (\lambda x. \text{poly } q \ x / \text{poly } p \ x) \ a = \text{jump-poly } q \ p \ a$

<proof>

lemma *jump-Im-divide-Re-0*:

assumes $\text{path } g \ \text{Re } (g \ x) \neq 0 \ 0 < x < 1$

shows $\text{jump } (\lambda t. \text{Im } (g \ t) / \text{Re } (g \ t)) \ x = 0$

<proof>

lemma *jumpF-im-divide-Re-0*:

assumes $\text{path } g \ \text{Re } (g \ x) \neq 0$

shows $\llbracket 0 \leq x; x < 1 \rrbracket \implies \text{jumpF } (\lambda t. \text{Im } (g \ t) / \text{Re } (g \ t)) \ (\text{at-right } x) = 0$

$\llbracket 0 < x; x \leq 1 \rrbracket \implies \text{jumpF } (\lambda t. \text{Im } (g \ t) / \text{Re } (g \ t)) \ (\text{at-left } x) = 0$

<proof>

lemma *jump-cong*:

assumes $x=y$ **and** $\text{eventually } (\lambda x. f \ x=g \ x) \ (\text{at } x)$

shows $\text{jump } f \ x = \text{jump } g \ y$

<proof>

lemma *jumpF-cong*:

assumes $F=G$ **and** $\text{eventually } (\lambda x. f \ x=g \ x) \ F$

shows $\text{jumpF } f \ F = \text{jumpF } g \ G$

<proof>

lemma *jump-at-left-at-right-eq*:

assumes $\text{isCont } f \ x$ **and** $f \ x \neq 0$ **and** $\text{sgnx-eq}:\text{sgnx } g \ (\text{at-left } x) = \text{sgnx } g \ (\text{at-right } x)$

shows $\text{jump } (\lambda t. f \ t/g \ t) \ x = 0$

<proof>

lemma *jumpF-pos-has-sgnx*:

assumes $\text{jumpF } f \ F > 0$

shows $(f \ \text{has-sgnx } 1) \ F$

<proof>

lemma *jumpF-neg-has-sgnx*:

assumes $\text{jumpF } f \ F < 0$
shows $(f \text{ has-sgn } x - 1) \ F$
 $\langle \text{proof} \rangle$

lemma *jumpF-IVT*:

fixes $f::\text{real} \Rightarrow \text{real}$ **and** $a \ b::\text{real}$

defines $\text{right} \equiv (\lambda(R::\text{real} \Rightarrow \text{real} \Rightarrow \text{bool}). R (\text{jumpF } f (\text{at-right } a)) \ 0$
 $\quad \vee (\text{continuous } (\text{at-right } a) \ f \wedge R (f \ a) \ 0))$

and

$\text{left} \equiv (\lambda(R::\text{real} \Rightarrow \text{real} \Rightarrow \text{bool}). R (\text{jumpF } f (\text{at-left } b)) \ 0$
 $\quad \vee (\text{continuous } (\text{at-left } b) \ f \wedge R (f \ b) \ 0))$

assumes $a < b$ **and** $\text{cont}:\text{continuous-on } \{a <..<b\}$ f **and**

$\text{right-left}:\text{right greater} \wedge \text{left less} \vee \text{right less} \wedge \text{left greater}$

shows $\exists x. a < x \wedge x < b \wedge f \ x = 0$

$\langle \text{proof} \rangle$

lemma *jumpF-eventually-const*:

assumes $\text{eventually } (\lambda x. f \ x = c) \ F \ F \neq \text{bot}$

shows $\text{jumpF } f \ F = 0$

$\langle \text{proof} \rangle$

lemma *jumpF-tan-comp*:

$\text{jumpF } (f \circ \text{tan}) (\text{at-right } x) = (\text{if } \cos \ x = 0$
 $\quad \text{then } \text{jumpF } f \ \text{at-bot} \ \text{else } \text{jumpF } f (\text{at-right } (\text{tan } x)))$

$\text{jumpF } (f \circ \text{tan}) (\text{at-left } x) = (\text{if } \cos \ x = 0$
 $\quad \text{then } \text{jumpF } f \ \text{at-top} \ \text{else } \text{jumpF } f (\text{at-left } (\text{tan } x)))$

$\langle \text{proof} \rangle$

5.6 Finite jumpFs over an interval

definition *finite-jumpFs*:: $(\text{real} \Rightarrow \text{real}) \Rightarrow \text{real} \Rightarrow \text{real} \Rightarrow \text{bool}$ **where**

$\text{finite-jumpFs } f \ a \ b = \text{finite } \{x. (\text{jumpF } f (\text{at-left } x) \neq 0 \vee \text{jumpF } f (\text{at-right } x) \neq 0) \wedge a \leq x \wedge x \leq b\}$

lemma *finite-jumpFs-linear-pos*:

assumes $c > 0$

shows $\text{finite-jumpFs } (f \circ (\lambda x. c * x + b)) \ lb \ ub \longleftrightarrow \text{finite-jumpFs } f \ (c * lb + b)$
 $(c * ub + b)$

$\langle \text{proof} \rangle$

lemma *finite-jumpFs-consts*:

$\text{finite-jumpFs } (\lambda \cdot . c) \ lb \ ub$

$\langle \text{proof} \rangle$

lemma *finite-jumpFs-combine*:

assumes $\text{finite-jumpFs } f \ a \ b$ $\text{finite-jumpFs } f \ b \ c$

shows $\text{finite-jumpFs } f \ a \ c$

$\langle \text{proof} \rangle$

lemma *finite-jumpFs-subE*:
assumes *finite-jumpFs f a b a ≤ a' b' ≤ b*
shows *finite-jumpFs f a' b'*
⟨*proof*⟩

lemma *finite-Psegments-Re-imp-jumpFs*:
assumes *finite-Psegments (λt. Re (g t - z) = 0) a b continuous-on {a..b} g*
shows *finite-jumpFs (λt. Im (g t - z)/Re (g t - z)) a b*
⟨*proof*⟩

lemma *finite-ReZ-segments-imp-jumpFs*:
assumes *finite-ReZ-segments g z path g*
shows *finite-jumpFs (λt. Im (g t - z)/Re (g t - z)) 0 1*
⟨*proof*⟩

5.7 *jumpF* at path ends

definition *jumpF-pathstart::(real ⇒ complex) ⇒ complex ⇒ real where*
jumpF-pathstart g z = jumpF (λt. Im(g t - z)/Re(g t - z)) (at-right 0)

definition *jumpF-pathfinish::(real ⇒ complex) ⇒ complex ⇒ real where*
jumpF-pathfinish g z = jumpF (λt. Im(g t - z)/Re(g t - z)) (at-left 1)

lemma *jumpF-pathstart-eq-0*:
assumes *path g Re(pathstart g) ≠ Re z*
shows *jumpF-pathstart g z = 0*
⟨*proof*⟩

lemma *jumpF-pathfinish-eq-0*:
assumes *path g Re(pathfinish g) ≠ Re z*
shows *jumpF-pathfinish g z = 0*
⟨*proof*⟩

lemma
shows *jumpF-pathfinish-reversepath: jumpF-pathfinish (reversepath g) z = jumpF-pathstart g z*
and *jumpF-pathstart-reversepath: jumpF-pathstart (reversepath g) z = jumpF-pathfinish g z*
⟨*proof*⟩

lemma *jumpF-pathstart-joinpaths[simp]*:
jumpF-pathstart (g1+++g2) z = jumpF-pathstart g1 z
⟨*proof*⟩

lemma *jumpF-pathfinish-joinpaths[simp]*:
jumpF-pathfinish (g1+++g2) z = jumpF-pathfinish g2 z
⟨*proof*⟩

5.8 Cauchy index

definition *cindex*::*real* \Rightarrow *real* \Rightarrow (*real* \Rightarrow *real*) \Rightarrow *int* **where**
cindex *a b f* = ($\sum x \in \{x. \text{jump } f \ x \neq 0 \wedge a < x \wedge x < b\}. \text{jump } f \ x$)

definition *cindexE*::*real* \Rightarrow *real* \Rightarrow (*real* \Rightarrow *real*) \Rightarrow *real* **where**
cindexE *a b f* = ($\sum x \in \{x. \text{jumpF } f \ (\text{at-right } x) \neq 0 \wedge a \leq x \wedge x < b\}. \text{jumpF } f \ (\text{at-right } x)$)
 $-$ ($\sum x \in \{x. \text{jumpF } f \ (\text{at-left } x) \neq 0 \wedge a < x \wedge x \leq b\}. \text{jumpF } f \ (\text{at-left } x)$)

definition *cindexE-ubd*::(*real* \Rightarrow *real*) \Rightarrow *real* **where**
cindexE-ubd *f* = ($\sum x \in \{x. \text{jumpF } f \ (\text{at-right } x) \neq 0\}. \text{jumpF } f \ (\text{at-right } x)$)
 $-$ ($\sum x \in \{x. \text{jumpF } f \ (\text{at-left } x) \neq 0\}. \text{jumpF } f \ (\text{at-left } x)$)

lemma *cindexE-empty*:
cindexE *a a f* = 0
 <proof>

lemma *cindex-const*: *cindex* *a b* ($\lambda-. c$) = 0
 <proof>

lemma *cindex-eq-cindex-poly*: *cindex* *a b* ($\lambda x. \text{poly } q \ x / \text{poly } p \ x$) = *cindex-poly* *a b q p*
 <proof>

lemma *cindex-combine*:
assumes *finite:finite* $\{x. \text{jump } f \ x \neq 0 \wedge a < x \wedge x < c\}$ **and** $a < b < c$
shows *cindex* *a c f* = *cindex* *a b f* + *jump* *f b* + *cindex* *b c f*
 <proof>

lemma *cindexE-combine*:
assumes *finite:finite-jumpFs* *f a c* **and** $a \leq b \leq c$
shows *cindexE* *a c f* = *cindexE* *a b f* + *cindexE* *b c f*
 <proof>

lemma *cindex-linear-comp*:
assumes $c \neq 0$
shows *cindex* *lb ub* (*f* o ($\lambda x. c * x + b$)) = (if $c > 0$
 then *cindex* ($c * lb + b$) ($c * ub + b$) *f*
 else $-$ *cindex* ($c * ub + b$) ($c * lb + b$) *f*)
 <proof>

lemma *cindexE-linear-comp*:
assumes $c \neq 0$
shows *cindexE* *lb ub* (*f* o ($\lambda x. c * x + b$)) = (if $c > 0$
 then *cindexE* ($c * lb + b$) ($c * ub + b$) *f*
 else $-$ *cindexE* ($c * ub + b$) ($c * lb + b$) *f*)
 <proof>

lemma *cindexE-cong*:

assumes *finite s* **and** *fg-eq*: $\bigwedge x. \llbracket a < x; x < b; x \notin s \rrbracket \implies f x = g x$
shows *cindexE a b f = cindexE a b g*

\langle *proof* \rangle

lemma *cindexE-constI*:

assumes $\bigwedge t. \llbracket a < t; t < b \rrbracket \implies f t = c$
shows *cindexE a b f = 0*

\langle *proof* \rangle

lemma *cindex-eq-cindexE-divide*:

fixes *f g*:*real* \Rightarrow *real*

defines *h* $\equiv (\lambda x. f x / g x)$

assumes *a < b* **and**

finite-fg: *finite* $\{x. (f x = 0 \vee g x = 0) \wedge a < x \wedge x \leq b\}$ **and**

g-imp-f: $\forall x \in \{a..b\}. g x = 0 \longrightarrow f x \neq 0$ **and**

f-cont: *continuous-on* $\{a..b\}$ *f* **and**

g-cont: *continuous-on* $\{a..b\}$ *g*

shows *cindexE a b h = jumpF h (at-right a) + cindex a b h - jumpF h (at-left b)*

\langle *proof* \rangle

5.9 Cauchy index along a path

definition *cindex-path*:*(real* \Rightarrow *complex*) \Rightarrow *complex* \Rightarrow *int* **where**

cindex-path g z = cindex 0 1 ($\lambda t. \text{Im } (g t - z) / \text{Re } (g t - z)$)

definition *cindex-pathE*:*(real* \Rightarrow *complex*) \Rightarrow *complex* \Rightarrow *real* **where**

cindex-pathE g z = cindexE 0 1 ($\lambda t. \text{Im } (g t - z) / \text{Re } (g t - z)$)

lemma *cindex-pathE-point*: *cindex-pathE (linepath a a) b = 0*

\langle *proof* \rangle

lemma *cindex-path-reversepath*:

cindex-path (reversepath g) z = - cindex-path g z

\langle *proof* \rangle

lemma *cindex-pathE-reversepath*: *cindex-pathE (reversepath g) z = -cindex-pathE g z*

\langle *proof* \rangle

lemma *cindex-pathE-reversepath'*: *cindex-pathE g z = -cindex-pathE (reversepath g) z*

\langle *proof* \rangle

lemma *cindex-pathE-joinpaths*:

assumes *g1*:*finite-ReZ-segments g1 z* **and** *g2*: *finite-ReZ-segments g2 z* **and**
path g1 path g2 pathfinish g1 = pathstart g2

shows $\text{cindex-pathE } (g1+++g2) z = \text{cindex-pathE } g1 z + \text{cindex-pathE } g2 z$
 ⟨proof⟩

lemma *cindex-pathE-constI*:
assumes $\bigwedge t. \llbracket 0 < t; t < 1 \rrbracket \implies g t = c$
shows $\text{cindex-pathE } g z = 0$
 ⟨proof⟩

lemma *cindex-pathE-subpath-combine*:
assumes *g:finite-ReZ-segments g* **and** *path g* **and**
 $0 \leq a \leq b \leq c \leq 1$
shows $\text{cindex-pathE } (\text{subpath } a \ b \ g) z + \text{cindex-pathE } (\text{subpath } b \ c \ g) z$
 $= \text{cindex-pathE } (\text{subpath } a \ c \ g) z$
 ⟨proof⟩

lemma *cindex-pathE-shiftpath*:
assumes *finite-ReZ-segments g z* $s \in \{0..1\}$ *path g* **and** *loop:pathfinish g = pathstart g*
shows $\text{cindex-pathE } (\text{shiftpath } s \ g) z = \text{cindex-pathE } g z$
 ⟨proof⟩

5.10 Cauchy's Index Theorem

theorem *winding-number-cindex-pathE-aux*:
fixes $g::\text{real} \Rightarrow \text{complex}$
assumes *finite-ReZ-segments g z* **and** *valid-path g z* $\notin \text{path-image } g$ **and**
 $\text{Re-ends:Re } (g \ 1) = \text{Re } z \ \text{Re } (g \ 0) = \text{Re } z$
shows $2 * \text{Re}(\text{winding-number } g \ z) = - \text{cindex-pathE } g \ z$
 ⟨proof⟩

theorem *winding-number-cindex-pathE*:
fixes $g::\text{real} \Rightarrow \text{complex}$
assumes *finite-ReZ-segments g z* **and** *valid-path g z* $\notin \text{path-image } g$ **and**
 $\text{loop: pathfinish } g = \text{pathstart } g$
shows $\text{winding-number } g \ z = - \text{cindex-pathE } g \ z / 2$
 ⟨proof⟩

REMARK: The usual statement of Cauchy's Index theorem (i.e. Analytic Theory of Polynomials (2002): Theorem 11.1.3) is about the equality between the number of polynomial roots and the Cauchy index, which is the joint application of $\llbracket \text{finite-ReZ-segments } ?g \ ?z; \text{valid-path } ?g; ?z \notin \text{path-image } ?g; \text{pathfinish } ?g = \text{pathstart } ?g \rrbracket \implies \text{winding-number } ?g \ ?z = \text{complex-of-real } (- \text{cindex-pathE } ?g \ ?z / 2)$ and $\llbracket \text{open } ?s; \text{connected } ?s; ?f \text{ holomorphic-on } ?s - ?poles; ?h \text{ holomorphic-on } ?s; \text{valid-path } ?g; \text{pathfinish } ?g = \text{pathstart } ?g; \text{path-image } ?g \subseteq ?s - \{w. ?f \ w = 0 \vee w \in ?poles\}; \forall z. z \notin ?s \longrightarrow \text{winding-number } ?g \ z = 0; \text{finite } \{w. ?f \ w = 0 \vee w \in ?poles\}; \forall p \in ?poles. \text{is-pole } ?f \ p \rrbracket \implies \text{contour-integral } ?g \ (\lambda x. \text{deriv } ?f \ x * ?h \ x / ?f \ x) = \text{complex-of-real } (2 * \text{pi}) * i * (\sum p \in \{w. ?f \ w = 0 \vee w \in ?poles\}. \text{winding-number } ?g \ p * ?h \ p * \text{of-int } (z \ \text{order } ?f \ p))$.

end

6 Evaluate winding numbers by calculating Cauchy indices

theory *Winding-Number-Eval* **imports**
 Cauchy-Index-Theorem
 HOL-Eisbach.Eisbach-Tools
begin

6.1 Misc

lemma *not-on-closed-segmentI*:
 fixes $z::'a::\text{euclidean-space}$
 assumes $\text{norm } (z - a) *_{\mathbb{R}} (b - z) \neq \text{norm } (b - z) *_{\mathbb{R}} (z - a)$
 shows $z \notin \text{closed-segment } a \ b$
 $\langle \text{proof} \rangle$

lemma *not-on-closed-segmentI-complex*:
 fixes $z::\text{complex}$
 assumes $(\text{Re } b - \text{Re } z) * (\text{Im } z - \text{Im } a) \neq (\text{Im } b - \text{Im } z) * (\text{Re } z - \text{Re } a)$
 shows $z \notin \text{closed-segment } a \ b$
 $\langle \text{proof} \rangle$

6.2 finite intersection with the two axes

definition *finite-axes-cross*:: $(\text{real} \Rightarrow \text{complex}) \Rightarrow \text{complex} \Rightarrow \text{bool}$ **where**
 $\text{finite-axes-cross } g \ z = \text{finite } \{t. (\text{Re } (g \ t - z) = 0 \vee \text{Im } (g \ t - z) = 0) \wedge 0 \leq t \wedge t \leq 1\}$

lemma *finite-cross-intros*:
 $\llbracket \text{Re } a \neq \text{Re } z \vee \text{Re } b \neq \text{Re } z; \text{Im } a \neq \text{Im } z \vee \text{Im } b \neq \text{Im } z \rrbracket \Longrightarrow \text{finite-axes-cross } (\text{linepath } a \ b) \ z$
 $\llbracket st \neq tt; r \neq 0 \rrbracket \Longrightarrow \text{finite-axes-cross } (\text{part-circlepath } z0 \ r \ st \ tt) \ z$
 $\llbracket \text{finite-axes-cross } g1 \ z; \text{finite-axes-cross } g2 \ z \rrbracket \Longrightarrow \text{finite-axes-cross } (g1 +++ g2) \ z$
 $\langle \text{proof} \rangle$

lemma *cindex-path-joinpaths*:
 assumes $\text{finite-axes-cross } g1 \ z \ \text{finite-axes-cross } g2 \ z$
 and $\text{path } g1 \ \text{path } g2 \ \text{pathfinish } g1 = \text{pathstart } g2 \ \text{pathfinish } g1 \neq z$
 shows $\text{cindex-path } (g1 +++ g2) \ z = \text{cindex-path } g1 \ z + \text{jumpF-pathstart } g2 \ z$
 $- \text{jumpF-pathfinish } g1 \ z + \text{cindex-path } g2 \ z$
 $\langle \text{proof} \rangle$

6.3 More lemmas related *cindex-pathE* / *jumpF-pathstart* / *jumpF-pathfinish*

lemma *cindex-pathE-linepath*:
 assumes $z \notin \text{closed-segment } a \ b$
 shows $\text{cindex-pathE } (\text{linepath } a \ b) \ z = ($

let $c1 = \text{Re } a - \text{Re } z$;
 $c2 = \text{Re } b - \text{Re } z$;
 $c3 = \text{Im } a * \text{Re } b + \text{Re } z * \text{Im } b + \text{Im } z * \text{Re } a - \text{Im } z * \text{Re } b - \text{Im } b * \text{Re } a - \text{Re } z * \text{Im } a$;
 $d1 = \text{Im } a - \text{Im } z$;
 $d2 = \text{Im } b - \text{Im } z$
 in if $(c1 > 0 \wedge c2 < 0) \vee (c1 < 0 \wedge c2 > 0)$ then
 (if $c3 > 0$ then 1 else -1)
 else
 (if $(c1 = 0 \leftrightarrow c2 \neq 0) \wedge (c1 = 0 \rightarrow d1 \neq 0) \wedge (c2 = 0 \rightarrow d2 \neq 0)$ then
 if $(c1 = 0 \wedge (c2 > 0 \leftrightarrow d1 > 0)) \vee (c2 = 0 \wedge (c1 > 0 \leftrightarrow d2 < 0))$ then
 1/2 else -1/2
 else 0))
 <proof>

lemma *cindex-path-linepath*:

assumes $z \notin \text{path-image } (\text{linepath } a \ b)$
 shows *cindex-path* (*linepath* $a \ b$) $z =$ (
 let $c1 = \text{Re}(a) - \text{Re}(z)$; $c2 = \text{Re}(b) - \text{Re}(z)$;
 $c3 = \text{Im}(a) * \text{Re}(b) + \text{Re}(z) * \text{Im}(b) + \text{Im}(z) * \text{Re}(a) - \text{Im}(z) * \text{Re}(b) - \text{Im}(b) * \text{Re}(a) - \text{Re}(z) * \text{Im}(a)$
 in if $(c1 > 0 \wedge c2 < 0) \vee (c1 < 0 \wedge c2 > 0)$ then (if $c3 > 0$ then 1 else -1) else 0)
 <proof>

lemma *cindex-pathE-part-circlepath*:

assumes $\text{cmod } (z - z0) \neq r$ and $r > 0$ $0 \leq st < tt$ $tt \leq 2 * \pi$
 shows *cindex-pathE* (*part-circlepath* $z \ r \ st \ tt$) $z0 =$ (
 if $|\text{Re } z - \text{Re } z0| < r$ then
 (let
 $\vartheta = \arccos ((\text{Re } z0 - \text{Re } z) / r)$;
 $\beta = 2 * \pi - \vartheta$
 in
 jumpF-pathstart (*part-circlepath* $z \ r \ st \ tt$) $z0$
 +
 (if $st < \vartheta \wedge \vartheta < tt$ then if $r * \sin \vartheta + \text{Im } z > \text{Im } z0$ then -1 else 1 else 0)
 +
 (if $st < \beta \wedge \beta < tt$ then if $r * \sin \beta + \text{Im } z > \text{Im } z0$ then 1 else -1 else 0)
 -
 jumpF-pathfinish (*part-circlepath* $z \ r \ st \ tt$) $z0$
)
 else
 if $|\text{Re } z - \text{Re } z0| = r$ then
 jumpF-pathstart (*part-circlepath* $z \ r \ st \ tt$) $z0$
 - *jumpF-pathfinish* (*part-circlepath* $z \ r \ st \ tt$) $z0$
 else 0
)
 <proof>

lemma *jumpF-pathstart-part-circlepath*:
assumes $st < tt$ $r > 0$ $cmod (z - z0) \neq r$
shows *jumpF-pathstart* (*part-circlepath* z r st tt) $z0 =$ (
 if $r * \cos st + \operatorname{Re} z - \operatorname{Re} z0 = 0$ then
 (let
 $\Delta = r * \sin st + \operatorname{Im} z - \operatorname{Im} z0$
 in
 if $(\sin st > 0 \vee \cos st = 1) \wedge \Delta < 0$
 $\vee (\sin st < 0 \vee \cos st = -1) \wedge \Delta > 0$ then
 $1/2$
 else
 $-1/2$)
 else 0)
 ⟨proof⟩

lemma *jumpF-pathfinish-part-circlepath*:
assumes $st < tt$ $r > 0$ $cmod (z - z0) \neq r$
shows *jumpF-pathfinish* (*part-circlepath* z r st tt) $z0 =$ (
 if $r * \cos tt + \operatorname{Re} z - \operatorname{Re} z0 = 0$ then
 (let
 $\Delta = r * \sin tt + \operatorname{Im} z - \operatorname{Im} z0$
 in
 if $(\sin tt > 0 \vee \cos tt = -1) \wedge \Delta < 0$
 $\vee (\sin tt < 0 \vee \cos tt = 1) \wedge \Delta > 0$ then
 $-1/2$
 else
 $1/2$)
 else 0)
 ⟨proof⟩

lemma
fixes $z0$ $z :: \text{complex}$ **and** $r :: \text{real}$
defines $upper \equiv cindex\text{-}pathE$ (*part-circlepath* z r 0 π) $z0$
and $lower \equiv cindex\text{-}pathE$ (*part-circlepath* z r π $(2 * \pi)$) $z0$
shows *cindex-pathE-circlepath-upper*:
 $\llbracket cmod (z0 - z) < r \rrbracket \implies upper = -1$
 $\llbracket \operatorname{Im} (z0 - z) > r; |\operatorname{Re} (z0 - z)| < r \rrbracket \implies upper = 1$
 $\llbracket \operatorname{Im} (z0 - z) < -r; |\operatorname{Re} (z0 - z)| < r \rrbracket \implies upper = -1$
 $\llbracket \operatorname{Re} (z0 - z) > r; r > 0 \rrbracket \implies upper = 0$
and *cindex-pathE-circlepath-lower*:
 $\llbracket cmod (z0 - z) < r \rrbracket \implies lower = -1$
 $\llbracket \operatorname{Im} (z0 - z) > r; |\operatorname{Re} (z0 - z)| < r \rrbracket \implies lower = -1$
 $\llbracket \operatorname{Im} (z0 - z) < -r; |\operatorname{Re} (z0 - z)| < r \rrbracket \implies lower = 1$
 $\llbracket \operatorname{Re} (z0 - z) > r; r > 0 \rrbracket \implies lower = 0$
 ⟨proof⟩

lemma *jumpF-pathstart-linepath*:
jumpF-pathstart (*linepath* a b) $z =$
 (if $\operatorname{Re} a = \operatorname{Re} z \wedge \operatorname{Im} a \neq \operatorname{Im} z \wedge \operatorname{Re} b \neq \operatorname{Re} a$ then

$\text{if } (Im\ a > Im\ z \wedge Re\ b > Re\ a) \vee (Im\ a < Im\ z \wedge Re\ b < Re\ a) \text{ then } 1/2 \text{ else}$
 $-1/2$
 $\text{else } 0)$
 <proof>

lemma *jumpF-pathfinish-linepath:*

$\text{jumpF-pathfinish } (\text{linepath } a\ b)\ z =$
 $(\text{if } Re\ b = Re\ z \wedge Im\ b \neq Im\ z \wedge Re\ b \neq Re\ a \text{ then}$
 $\text{if } (Im\ b > Im\ z \wedge Re\ a > Re\ b) \vee (Im\ b < Im\ z \wedge Re\ a < Re\ b) \text{ then } 1/2 \text{ else}$
 $-1/2$
 $\text{else } 0)$
 <proof>

6.4 Setting up the method for evaluating winding numbers

lemma *pathfinish-pathstart-partcirclepath-simps:*

$\text{pathstart } (\text{part-circlepath } z0\ r\ (3*\pi/2)\ tt) = z0 - \text{Complex } 0\ r$
 $\text{pathstart } (\text{part-circlepath } z0\ r\ (2*\pi)\ tt) = z0 + r$
 $\text{pathfinish } (\text{part-circlepath } z0\ r\ st\ (3*\pi/2)) = z0 - \text{Complex } 0\ r$
 $\text{pathfinish } (\text{part-circlepath } z0\ r\ st\ (2*\pi)) = z0 + r$
 $\text{pathstart } (\text{part-circlepath } z0\ r\ 0\ tt) = z0 + r$
 $\text{pathstart } (\text{part-circlepath } z0\ r\ (\pi/2)\ tt) = z0 + \text{Complex } 0\ r$
 $\text{pathstart } (\text{part-circlepath } z0\ r\ (\pi)\ tt) = z0 - r$
 $\text{pathfinish } (\text{part-circlepath } z0\ r\ st\ 0) = z0 + r$
 $\text{pathfinish } (\text{part-circlepath } z0\ r\ st\ (\pi/2)) = z0 + \text{Complex } 0\ r$
 $\text{pathfinish } (\text{part-circlepath } z0\ r\ st\ (\pi)) = z0 - r$
 <proof>

lemma *winding-eq-intro:*

$\text{finite-ReZ-segments } g\ z \implies$
 $\text{valid-path } g \implies$
 $z \notin \text{path-image } g \implies$
 $\text{pathfinish } g = \text{pathstart } g \implies$
 $-\text{of-real}(\text{cindex-pathE } g\ z) = 2*n \implies$
 $\text{winding-number } g\ z = (n::\text{complex})$
 <proof>

named-theorems *winding-intros and winding-simps*

lemmas [*winding-intros*] =

finite-ReZ-segments-joinpaths
valid-path-join
path-join-imp
not-in-path-image-join

lemmas [*winding-simps*] =

finite-ReZ-segments-linepath
finite-ReZ-segments-part-circlepath
jumpF-pathfinish-joinpaths

```

jumpF-pathstart-joinpaths
pathfinish-linepath
pathstart-linepath
pathfinish-join
pathstart-join
valid-path-linepath
valid-path-part-circlepath
path-part-circlepath
Re-complex-of-real
Im-complex-of-real
of-real-linepath
pathfinish-pathstart-partcirclepath-simps

```

```

method rep-subst =
  (subst cindex-pathE-joinpaths; rep-subst)?

```

The method "eval_winding" $1::'a$ will try to simplify of the form *winding-number* $g z = n$ where n is an integer and g is a closed path comprised of *linepath*, *part-circlepath* and (+++).

Suppose $g = l1 +++ l2$, usually, the key behind the success of this framework is whether we can prove $z \notin \text{path-image } l1$, $z \notin \text{path-image } l2$ and calculate *cindex-pathE* $l1 z$ and *cindex-pathE* $l2 z$.

```

method eval-winding =
  ((rule-tac winding-eq-intro;
   rep-subst
  )
  , auto simp only:winding-simps del:notI intro!:winding-intros
  , tactic <distinct-subgoals-tac>)

```

end

7 Some examples of applying the method winding_eval

```

theory Winding-Number-Eval-Examples imports Winding-Number-Eval
begin

```

```

lemma example1:
  assumes  $R > 1$ 
  shows winding-number (part-circlepath 0 R 0 pi +++ linepath (-R) R) i = 1
  <proof>

```

```

lemma example2:
  assumes  $R > 1$ 
  shows winding-number (part-circlepath 0 R 0 pi +++ linepath (-R) R) (-i) =
  0
  <proof>

```

```

lemma example3:
  fixes lb ub z :: complex
  defines rec ≡ linepath lb (Complex (Re ub) (Im lb)) +++ linepath (Complex
(Re ub) (Im lb)) ub
          +++ linepath ub (Complex (Re lb) (Im ub)) +++ linepath (Complex
(Re lb) (Im ub)) lb
  assumes order-asms:Re lb < Re z Re z < Re ub Im lb < Im z Im z < Im ub
  shows winding-number rec z = 1
  ⟨proof⟩

end

```

8 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] M. Eisermann. The fundamental theorem of algebra made effective: An elementary real-algebraic proof via Sturm chains. *American Mathematical Monthly*, 119(9):715–752, 2012.
- [2] Q. I. Rahman and G. Schmeisser. *Analytic theory of polynomials*. Number 26. Oxford University Press, 2002.