The Poincaré-Bendixson Theorem

Fabian Immler and Yong Kiam Tan

December 14, 2021

Contents

1 Additions to HOL-Analysis

theory *Analysis-Misc*
imports
Ordinary-Differential-Equations.ODE-Analysis
begin

1.1 Unsorted Lemmas (TODO: sort!)

lemma *uminus-uminus-image*: $uminus\ `\ uminus\ `\ S = S$
for $S::'r::ab\text{-}group\text{-}add\ set$
⟨*proof*⟩

lemma *in-uminus-image-iff*[*simp*]: $x \in uminus\ `\ S \longleftrightarrow - x \in S$
for $S::'r::ab\text{-}group\text{-}add\ set$
⟨*proof*⟩

lemma *closed-subsegmentI*:
$w + t *_R (z - w) \in \{x{-}{-}y\}$
if $w \in \{x -- y\}$ $z \in \{x -- y\}$ **and** t: $0 \le t$ $t \le 1$
⟨*proof*⟩

lemma *tendsto-minus-cancel-right*: $((\lambda x. -g\ x) \longrightarrow l)\ F \longleftrightarrow (g \longrightarrow -l)\ F$
— cf $(?f \longrightarrow -\ ?y)\ ?F = ((\lambda x. -\ ?f\ x) \longrightarrow\ ?y)\ ?F$
for $g::\text{-} \Rightarrow 'b::topological\text{-}group\text{-}add$
⟨*proof*⟩

lemma *tendsto-nhds-continuousI*: $(f \longrightarrow l)\ (nhds\ x)$ **if** $(f \longrightarrow l)\ (at\ x)$ $f\ x = l$
— TODO: the assumption is continuity of f at x
⟨*proof*⟩

lemma *inj-composeD*:
assumes $inj\ (\lambda x.\ g\ (t\ x))$
shows $inj\ t$
⟨*proof*⟩

lemma *compact-sequentialE*:
fixes $S\ T::'a::first\text{-}countable\text{-}topology\ set$

assumes *compact S*
assumes *infinite T*
assumes $T \subseteq S$
obtains t::$nat \Rightarrow 'a$ **and** l::$'a$
where $\bigwedge n.\ t\ n \in T$ $\bigwedge n.\ t\ n \neq l$ $t \longrightarrow l$ $l \in S$
⟨*proof*⟩

lemma *infinite-countable-subsetE*:
fixes S::$'a$ *set*
assumes *infinite S*
obtains g::$nat \Rightarrow 'a$ **where** *inj g range g* $\subseteq S$
⟨*proof*⟩

lemma *real-quad-ge*: $2 * (an * bn) \leq an * an + bn * bn$ **for** *an bn*::*real*
⟨*proof*⟩

lemma *inner-quad-ge*: $2 * (a \cdot b) \leq a \cdot a + b \cdot b$
for a b::$'a$::*euclidean-space*— generalize?
⟨*proof*⟩

lemma *inner-quad-gt*: $2 * (a \cdot b) < a \cdot a + b \cdot b$
if $a \neq b$
for a b::$'a$::*euclidean-space*— generalize?
⟨*proof*⟩

lemma *closed-segment-line-hyperplanes*:
$\{a -- b\} = range\ (\lambda u.\ a + u *_R (b - a)) \cap \{x.\ a \cdot (b - a) \leq x \cdot (b - a) \wedge x \cdot (b - a) \leq b \cdot (b - a)\}$
if $a \neq b$
for a b::$'a$::*euclidean-space*
⟨*proof*⟩

lemma *open-segment-line-hyperplanes*:
$\{a <--< b\} = range\ (\lambda u.\ a + u *_R (b - a)) \cap \{x.\ a \cdot (b - a) < x \cdot (b - a) \wedge x \cdot (b - a) < b \cdot (b - a)\}$
if $a \neq b$
for a b::$'a$::*euclidean-space*
⟨*proof*⟩

lemma *at-within-interior*: *NO-MATCH UNIV S* $\Longrightarrow x \in$ *interior S* $\Longrightarrow$ *at x within S* $=$ *at x*
⟨*proof*⟩

lemma *tendsto-at-topI*:
$(f \longrightarrow l)$ *at-top* **if** $\bigwedge e.\ 0 < e \Longrightarrow \exists x0.\ \forall x \geq x0.\ dist\ (f\ x)\ l < e$
for f::$'a$::*linorder-topology* $\Rightarrow$ $'b$::*metric-space*
⟨*proof*⟩

lemma *tendsto-at-topE*:

fixes f::$'a$::*linorder-topology* $\Rightarrow$ $'b$::*metric-space*
assumes $(f \longrightarrow l)$ *at-top*
assumes $e > 0$
obtains $x0$ **where** $\bigwedge x.\ x \geq x0 \Longrightarrow dist\ (f\ x)\ l < e$
⟨*proof*⟩
lemma *tendsto-at-top-iff*: $(f \longrightarrow l)$ *at-top* $\longleftrightarrow (\forall\, e{>}0.\ \exists\, x0.\ \forall\, x{\geq}x0.\ dist\ (f\ x)\ l < e)$
for f::$'a$::*linorder-topology* $\Rightarrow$ $'b$::*metric-space*
⟨*proof*⟩

lemma *tendsto-at-top-eq-left*:
fixes $f\ g$::$'a$::*linorder-topology* $\Rightarrow$ $'b$::*metric-space*
assumes $(f \longrightarrow l)$ *at-top*
assumes $\bigwedge x.\ x \geq x0 \Longrightarrow f\ x = g\ x$
shows $(g \longrightarrow l)$ *at-top*
⟨*proof*⟩

lemma *lim-divide-n*: $(\lambda x.\ e\ /\ real\ x) \longrightarrow 0$
⟨*proof*⟩

definition *at-top-within* :: $('a$::*order*$)$ *set* $\Rightarrow$ $'a$ *filter*
where *at-top-within* s = (*INF* $k \in s$. *principal* $(\{k\ ..\} \cap s)$)

lemma *at-top-within-at-top*[*simp*]:
shows *at-top-within UNIV* = *at-top*
⟨*proof*⟩

lemma *at-top-within-empty*[*simp*]:
shows *at-top-within* $\{\}$ = *top*
⟨*proof*⟩

definition *nhds-set* X = (*INF* $S{\in}\{S.\ open\ S \wedge X \subseteq S\}$. *principal* S)

lemma *eventually-nhds-set*:
$(\forall_F\ x\ in\ nhds\text{-}set\ X.\ P\ x) \longleftrightarrow (\exists\, S.\ open\ S \wedge X \subseteq S \wedge (\forall\, x{\in}S.\ P\ x))$
⟨*proof*⟩

term *filterlim f* (*nhds-set* (*frontier* X)) F — f tends to the boundary of X?

somewhat inspired by *?l islimpt range ?f* $\Longrightarrow \exists\, r.$ *strict-mono* $r \wedge (?f \circ r) \longrightarrow\ ?l$ and its dependencies. The class constraints seem somewhat arbitrary, perhaps this can be generalized in some way.

lemma *limpt-closed-imp-exploding-subsequence*:— TODO: improve name?!
fixes f::$'a$::$\{$*heine-borel*,*real-normed-vector*$\}$ $\Rightarrow$ $'b$::$\{$*first-countable-topology*, *t2-space*$\}$
assumes *cont*[*THEN continuous-on-compose2*, *continuous-intros*]: *continuous-on* $T\ f$
assumes *closed*: *closed* T
assumes *bound*: $\bigwedge t.\ t \in T \Longrightarrow f\ t \neq l$
assumes *limpt*: l *islimpt* $(f\ `\ T)$

obtains s **where**
$(f \circ s) \longrightarrow l$
$\bigwedge i.\ s\ i \in T$
$\bigwedge C.\ compact\ C \Longrightarrow C \subseteq T \Longrightarrow \forall_F\ i\ in\ sequentially.\ s\ i \notin C$
$\langle proof \rangle$

lemma *Inf-islimpt*: $bdd\text{-}below\ S \Longrightarrow Inf\ S \notin S \Longrightarrow S \neq \{\} \Longrightarrow Inf\ S\ islimpt\ S$ **for** S::*real set*
$\langle proof \rangle$

context *linorder*
begin

HOL-analysis doesn't seem to have these, maybe they were never needed. Some variants are around $\{?a..?b\} \cap \{?c..?d\} = \{max\ ?a\ ?c..min\ ?b\ ?d\}$, but with old-style naming conventions. Change to the "modern" I.. convention there?

lemma *Int-Ico*[*simp*]:
shows $\{a..\} \cap \{b..\} = \{max\ a\ b\ ..\}$
$\langle proof \rangle$

lemma *Int-Ici-Ico*[*simp*]:
shows $\{a..\} \cap \{b..<c\} = \{max\ a\ b\ ..<c\}$
$\langle proof \rangle$

lemma *Int-Ico-Ici*[*simp*]:
shows $\{a..<c\} \cap \{b..\} = \{max\ a\ b\ ..<c\}$
$\langle proof \rangle$

lemma *subset-Ico-iff*[*simp*]:
$\{a..<b\} \subseteq \{c..<b\} \longleftrightarrow b \leq a \vee c \leq a$
$\langle proof \rangle$

lemma *Ico-subset-Ioo-iff*[*simp*]:
$\{a..<b\} \subseteq \{c<..<b\} \longleftrightarrow b \leq a \vee c < a$
$\langle proof \rangle$

lemma *Icc-Un-Ici*[*simp*]:
shows $\{a..b\} \cup \{b..\} = \{min\ a\ b..\}$
$\langle proof \rangle$

end

lemma *at-top-within-at-top-unbounded-right*:
fixes a::$'a$::*linorder*
shows $at\text{-}top\text{-}within\ \{a..\} = at\text{-}top$
$\langle proof \rangle$

lemma *at-top-within-at-top-unbounded-rightI*:

fixes $a::'a::linorder$
assumes $\{a..\} \subseteq s$
shows *at-top-within* s = *at-top*
⟨*proof*⟩

lemma *at-top-within-at-top-bounded-right*:
fixes $a\ b::'a::\{dense\text{-}order,linorder\text{-}topology\}$
assumes $a < b$
shows *at-top-within* $\{a..<b\}$ = *at-left* b
⟨*proof*⟩

lemma *at-top-within-at-top-bounded-right'*:
fixes $a\ b::'a::\{dense\text{-}order,linorder\text{-}topology\}$
assumes $a < b$
shows *at-top-within* $\{..<b\}$ = *at-left* b
⟨*proof*⟩

lemma *eventually-at-top-within-linorder*:
assumes $sn{:}s \neq \{\}$
shows *eventually* P (*at-top-within* s) $\longleftrightarrow$ $(\exists x0::'a::\{linorder\text{-}topology\} \in s.\ \forall x \geq x0.\ x \in s \longrightarrow P\ x)$
⟨*proof*⟩

lemma *tendsto-at-top-withinI*:
fixes $f::'a::linorder\text{-}topology \Rightarrow 'b::metric\text{-}space$
assumes $s \neq \{\}$
assumes $\bigwedge e.\ 0 < e \Longrightarrow \exists x0 \in s.\ \forall x \in \{x0..\} \cap s.\ dist\ (f\ x)\ l < e$
shows $(f \longrightarrow l)$ (*at-top-within* s)
⟨*proof*⟩

lemma *tendsto-at-top-withinE*:
fixes $f::'a::linorder\text{-}topology \Rightarrow 'b::metric\text{-}space$
assumes $s \neq \{\}$
assumes $(f \longrightarrow l)$ (*at-top-within* s)
assumes $e > 0$
obtains $x0$ **where** $x0 \in s\ \bigwedge x.\ x \in \{x0..\} \cap s \Longrightarrow dist\ (f\ x)\ l < e$
⟨*proof*⟩

lemma *tendsto-at-top-within-iff*:
fixes $f::'a::linorder\text{-}topology \Rightarrow 'b::metric\text{-}space$
assumes $s \neq \{\}$
shows $(f \longrightarrow l)$ (*at-top-within* s) $\longleftrightarrow$ $(\forall e>0.\ \exists x0 \in s.\ \forall x \in \{x0..\} \cap s.\ dist\ (f\ x)\ l < e)$
⟨*proof*⟩

lemma *filterlim-at-top-at-top-within-bounded-right*:
fixes $a\ b::'a::\{dense\text{-}order,linorder\text{-}topology\}$
fixes $f::'a \Rightarrow real$
assumes $a < b$

shows *filterlim f at-top* (*at-top-within* {..<*b*}) = (*f* $\longrightarrow$ ∞) (*at-left b*)
⟨*proof*⟩

Extract a sequence (going to infinity) bounded away from l

lemma *not-tendsto-frequentlyE*:
assumes $\neg$((*f* $\longrightarrow$ *l*) *F*)
obtains *S* **where** *open S l* $\in$ *S* $\exists_F$ *x in F*. *f x* $\notin$ *S*
⟨*proof*⟩

lemma *not-tendsto-frequently-metricE*:
assumes $\neg$((*f* $\longrightarrow$ *l*) *F*)
obtains *e* **where** *e* > *0* $\exists_F$ *x in F*. *e* $\le$ *dist* (*f x*) *l*
⟨*proof*⟩

lemma *eventually-frequently-conj*: *frequently P F* $\Longrightarrow$ *eventually Q F* $\Longrightarrow$ *frequently* (λ*x*. *P x* $\wedge$ *Q x*) *F*
⟨*proof*⟩

lemma *frequently-at-top*:
($\exists_F$ *t in at-top*. *P t*) $\longleftrightarrow$ ($\forall$ *t0*. $\exists$ *t*>*t0*. *P t*)
for *P*::′*a*::{*linorder*,*no-top*}$\Rightarrow$*bool*
⟨*proof*⟩

lemma *frequently-at-topE*:
fixes *P*::*nat* $\Rightarrow$ ′*a*::{*linorder*,*no-top*}$\Rightarrow$-
assumes *freq*[*rule-format*]: $\forall$ *n*. $\exists_F$ *a in at-top*. *P n a*
obtains *s*::*nat*$\Rightarrow$′*a*
where $\bigwedge$*i*. *P i* (*s i*) *strict-mono s*
⟨*proof*⟩

lemma *frequently-at-topE*′:
fixes *P*::*nat* $\Rightarrow$ ′*a*::{*linorder*,*no-top*}$\Rightarrow$-
assumes *freq*[*rule-format*]: $\forall$ *n*. $\exists_F$ *a in at-top*. *P n a*
and *g*: *filterlim g at-top sequentially*
obtains *s*::*nat*$\Rightarrow$′*a*
where $\bigwedge$*i*. *P i* (*s i*) *strict-mono s* $\bigwedge$*n*. *g n* $\le$ *s n*
⟨*proof*⟩

lemma *frequently-at-top-at-topE*:
fixes *P*::*nat* $\Rightarrow$ ′*a*::{*linorder*,*no-top*}$\Rightarrow$- **and** *g*::*nat*$\Rightarrow$′*a*
assumes $\forall$ *n*. $\exists_F$ *a in at-top*. *P n a filterlim g at-top sequentially*
obtains *s*::*nat*$\Rightarrow$′*a*
where $\bigwedge$*i*. *P i* (*s i*) *filterlim s at-top sequentially*
⟨*proof*⟩

lemma *not-tendsto-convergent-seq*:
fixes *f*::*real* $\Rightarrow$ ′*a*::*metric-space*
assumes *X*: *compact* (*X*::′*a set*)

assumes *im*: $\bigwedge x.\ x \geq 0 \Longrightarrow f\ x \in X$
assumes *nl*: $\neg\ ((f \longrightarrow (l::'a))\ \textit{at-top})$
obtains *s k* **where**
$k \in X\ k \neq l\ (f \circ s) \longrightarrow k\ \textit{strict-mono}\ s\ \forall n.\ s\ n \geq n$
⟨*proof*⟩

lemma *harmonic-bound*:
shows $1\ /\ 2\ \widehat{}\ (Suc\ n) < 1\ /\ real\ (Suc\ n)$
⟨*proof*⟩

lemma *INF-bounded-imp-convergent-seq*:
fixes $f::real \Rightarrow real$
assumes *cont*: *continuous-on* $\{a..\}\ f$
assumes *bound*: $\bigwedge t.\ t \geq a \Longrightarrow f\ t > l$
assumes *inf*: $(INF\ t{\in}\{a..\}.\ f\ t) = l$
obtains *s* **where**
$(f \circ s) \longrightarrow l$
$\bigwedge i.\ s\ i \in \{a..\}$
filterlim s at-top sequentially
⟨*proof*⟩

lemma *filterlim-at-top-strict-mono*:
fixes $s :: \text{-} \Rightarrow 'a::linorder$
fixes $r :: nat \Rightarrow \text{-}$
assumes *strict-mono s*
assumes *strict-mono r*
assumes *filterlim s at-top F*
shows *filterlim* $(s \circ r)$ *at-top F*
⟨*proof*⟩

lemma *LIMSEQ-lb*:
assumes *fl*: $s \longrightarrow (l::real)$
assumes *u*: $l < u$
shows $\exists n0.\ \forall n{\geq}n0.\ s\ n < u$
⟨*proof*⟩

lemma *filterlim-at-top-choose-lower*:
assumes *filterlim s at-top sequentially*
assumes $(f \circ s) \longrightarrow l$
obtains *t* **where**
filterlim t at-top sequentially
$(f \circ t) \longrightarrow l$
$\forall n.\ t\ n \geq (b::real)$
⟨*proof*⟩

lemma *frequently-at-top-realE*:
fixes $P::nat \Rightarrow real \Rightarrow bool$

assumes $\forall\, n.\ \exists_F\ t$ *in at-top*. $P\ n\ t$
obtains s::*nat*⇒*real*
where $\bigwedge i.\ P\ i\ (s\ i)$ *filterlim s at-top at-top*
⟨*proof*⟩

lemma *approachable-sequenceE*:
fixes f::*real* ⇒ *'a*::*metric-space*
assumes $\bigwedge t\ e.\ 0 \leq t \Longrightarrow 0 < e \Longrightarrow \exists\, tt \geq t.\ dist\ (f\ tt)\ p < e$
obtains s **where** *filterlim s at-top sequentially* $(f \circ s) \longlongrightarrow p$
⟨*proof*⟩

lemma *mono-inc-bdd-above-has-limit-at-topI*:
fixes f::*real* ⇒ *real*
assumes *mono f*
assumes $\bigwedge x.\ f\ x \leq u$
shows $\exists\, l.\ (f \longrightarrow l)$ *at-top*
⟨*proof*⟩

lemma *gen-mono-inc-bdd-above-has-limit-at-topI*:
fixes f::*real* ⇒ *real*
assumes $\bigwedge x\ y.\ x \geq b \Longrightarrow x \leq y \Longrightarrow f\ x \leq f\ y$
assumes $\bigwedge x.\ x \geq b \Longrightarrow f\ x \leq u$
shows $\exists\, l.\ (f \longrightarrow l)$ *at-top*
⟨*proof*⟩

lemma *gen-mono-dec-bdd-below-has-limit-at-topI*:
fixes f::*real* ⇒ *real*
assumes $\bigwedge x\ y.\ x \geq b \Longrightarrow x \leq y \Longrightarrow f\ x \geq f\ y$
assumes $\bigwedge x.\ x \geq b \Longrightarrow f\ x \geq u$
shows $\exists\, l.\ (f \longrightarrow l)$ *at-top*
⟨*proof*⟩

lemma *infdist-closed*:
shows *closed* $(\{z.\ infdist\ z\ S \geq e\})$
⟨*proof*⟩

lemma *LIMSEQ-norm-0-pow*:
assumes $k > 0\ b > 1$
assumes $\bigwedge n$::*nat*. $norm\ (s\ n) \leq k\ /\ b\hat{\ }n$
shows $s \longrightarrow 0$
⟨*proof*⟩

lemma *filterlim-apply-filtermap*:
assumes g: *filterlim g G F*
shows *filterlim* $(\lambda x.\ m\ (g\ x))$ $($*filtermap m G*$)$ F
⟨*proof*⟩

lemma *eventually-at-right-field-le*:

$eventually\ P\ (at\text{-}right\ x) \longleftrightarrow (\exists\, b{>}x.\ \forall\, y{>}x.\ y \le b \longrightarrow P\ y)$
for $x :: 'a::\{linordered\text{-}field,\ linorder\text{-}topology\}$
$\langle proof \rangle$

1.2 indexing euclidean space with natural numbers

definition $nth\text{-}eucl :: 'a::executable\text{-}euclidean\text{-}space \Rightarrow nat \Rightarrow real$ **where**
$nth\text{-}eucl\ x\ i = x \cdot (Basis\text{-}list\ !\ i)$
— TODO: why is that and some sort of *lambda-eucl* nowhere available?
definition $lambda\text{-}eucl :: (nat \Rightarrow real) \Rightarrow 'a::executable\text{-}euclidean\text{-}space$ **where**
$lambda\text{-}eucl\ (f::nat{\Rightarrow}real) = (\sum i{<}DIM('a).\ f\ i *_R Basis\text{-}list\ !\ i)$

lemma *eucl-eq-iff*: $x = y \longleftrightarrow (\forall\, i{<}DIM('a).\ nth\text{-}eucl\ x\ i = nth\text{-}eucl\ y\ i)$
for $x\ y::'a::executable\text{-}euclidean\text{-}space$
$\langle proof \rangle$

bundle *eucl-notation* **begin**
notation *nth-eucl* (**infixl** $\$_e$ *90*)
end
bundle *no-eucl-notation* **begin**
no-notation *nth-eucl* (**infixl** $\$_e$ *90*)
end

unbundle *eucl-notation*

lemma *eucl-of-list-eucl-nth*:
$(eucl\text{-}of\text{-}list\ xs::'a)\ \$_e\ i = xs\ !\ i$
if $length\ xs = DIM('a::executable\text{-}euclidean\text{-}space)$
$i < DIM('a)$
$\langle proof \rangle$

lemma *eucl-of-list-inner*:
$(eucl\text{-}of\text{-}list\ xs::'a) \cdot eucl\text{-}of\text{-}list\ ys = (\sum (x,y){\leftarrow}zip\ xs\ ys.\ x * y)$
if $length\ xs = DIM('a::executable\text{-}euclidean\text{-}space)$
$length\ ys = DIM('a::executable\text{-}euclidean\text{-}space)$
$\langle proof \rangle$

lemma *self-eq-eucl-of-list*: $x = eucl\text{-}of\text{-}list\ (map\ (\lambda i.\ x\ \$_e\ i)\ [0..{<}DIM('a)])$
for $x::'a::executable\text{-}euclidean\text{-}space$
$\langle proof \rangle$

lemma *inner-nth-eucl*: $x \cdot y = (\sum i{<}DIM('a).\ x\ \$_e\ i * y\ \$_e\ i)$
for $x\ y::'a::executable\text{-}euclidean\text{-}space$
$\langle proof \rangle$

lemma *norm-nth-eucl*: $norm\ x = L2\text{-}set\ (\lambda i.\ x\ \$_e\ i)\ \{..{<}DIM('a)\}$
for $x::'a::executable\text{-}euclidean\text{-}space$
$\langle proof \rangle$

lemma *plus-nth-eucl*: $(x + y)\ \$_e\ i = x\ \$_e\ i + y\ \$_e\ i$
and *minus-nth-eucl*: $(x - y)\ \$_e\ i = x\ \$_e\ i - y\ \$_e\ i$
and *uminus-nth-eucl*: $(-x)\ \$_e\ i = -\ x\ \$_e\ i$
and *scaleR-nth-eucl*: $(c *_R x)\ \$_e\ i = c *_R (x\ \$_e\ i)$
⟨*proof*⟩

lemma *inf-nth-eucl*: $inf\ x\ y\ \$_e\ i = min\ (x\ \$_e\ i)\ (y\ \$_e\ i)$
if $i < DIM('a)$
for $x{::}'a{::}executable\text{-}euclidean\text{-}space$
⟨*proof*⟩
lemma *sup-nth-eucl*: $sup\ x\ y\ \$_e\ i = max\ (x\ \$_e\ i)\ (y\ \$_e\ i)$
if $i < DIM('a)$
for $x{::}'a{::}executable\text{-}euclidean\text{-}space$
⟨*proof*⟩

lemma *le-iff-le-nth-eucl*: $x \le y \longleftrightarrow (\forall\, i{<}DIM('a).\ (x\ \$_e\ i) \le (y\ \$_e\ i))$
for $x{::}'a{::}executable\text{-}euclidean\text{-}space$
⟨*proof*⟩

lemma *eucl-less-iff-less-nth-eucl*: $eucl\text{-}less\ x\ y \longleftrightarrow (\forall\, i{<}DIM('a).\ (x\ \$_e\ i) < (y\ \$_e\ i))$
for $x{::}'a{::}executable\text{-}euclidean\text{-}space$
⟨*proof*⟩

lemma *continuous-on-nth-eucl*[*continuous-intros*]:
$continuous\text{-}on\ X\ (\lambda x.\ f\ x\ \$_e\ i)$
if $continuous\text{-}on\ X\ f$
⟨*proof*⟩

1.3 derivatives

lemma *eventually-at-ne*[*intro*, *simp*]: $\forall_F\ x\ in\ at\ x0.\ x \ne x0$
⟨*proof*⟩

lemma *has-vector-derivative-withinD*:
fixes $f{::}real \Rightarrow 'b{::}euclidean\text{-}space$
assumes $(f\ has\text{-}vector\text{-}derivative\ f')\ (at\ x0\ within\ S)$
shows $((\lambda x.\ (f\ x - f\ x0)\ /_R\ (x - x0)) \longrightarrow f')\ (at\ x0\ within\ S)$
⟨*proof*⟩

A *path-connected* set S entering both T and $-T$ must cross the frontier of T

lemma *path-connected-frontier*:
fixes $S :: 'a{::}real\text{-}normed\text{-}vector\ set$
assumes $path\text{-}connected\ S$
assumes $S \cap T \ne \{\}$
assumes $S \cap -T \ne \{\}$
obtains s **where** $s \in S\ s \in frontier\ T$

⟨proof⟩

lemma *path-connected-not-frontier-subset*:
fixes S :: $'a$::*real-normed-vector set*
assumes *path-connected* S
assumes $S \cap T \neq \{\}$
assumes $S \cap$ *frontier* $T = \{\}$
shows $S \subseteq T$
⟨proof⟩

lemma *compact-attains-bounds*:
fixes f::$'a$::*topological-space* $\Rightarrow$ $'b$::*linorder-topology*
assumes *compact*: *compact* S
assumes *ne*: $S \neq \{\}$
assumes *cont*: *continuous-on* S f
obtains l u **where** $l \in S$ $u \in S$ $\bigwedge x.\ x \in S \Longrightarrow f\ x \in \{f\ l\ ..\ f\ u\}$
⟨proof⟩

lemma *uniform-limit-const*[*uniform-limit-intros*]:
uniform-limit S $(\lambda x\ y.\ f\ x)$ $(\lambda\text{-}.\ l)$ F **if** $(f \longrightarrow l)$ F
⟨proof⟩

1.4 Segments

closed-segment throws away the order that our intuition keeps

definition *line*::$'a$::*real-vector* $\Rightarrow$ $'a$ $\Rightarrow$ *real* $\Rightarrow$ $'a$
$(\{\text{-} -- \text{-}\}_\text{-})$
where $\{a -- b\}_u = a + u *_R (b - a)$

abbreviation *line-image* a b U $\equiv(\lambda u.\ \{a -- b\}_u)$ ‘ U
notation *line-image* $(\{\text{-} -- \text{-}\}\text{‘}_\text{-})$

lemma *in-closed-segment-iff-line*: $x \in \{a -- b\} \longleftrightarrow (\exists\, c\in\{0..1\}.\ x = line\ a\ b\ c)$
⟨proof⟩

lemma *in-open-segment-iff-line*: $x \in \{a <--< b\} \longleftrightarrow (\exists\, c\in\{0<..<1\}.\ a \neq b \wedge x = line\ a\ b\ c)$
⟨proof⟩

lemma *line-convex-combination1*: $(1 - u) *_R line\ a\ b\ i + u *_R b = line\ a\ b\ (i + u - i * u)$
⟨proof⟩

lemma *line-convex-combination2*: $(1 - u) *_R a + u *_R line\ a\ b\ i = line\ a\ b\ (i{*}u)$
⟨proof⟩

lemma *line-convex-combination12*: $(1 - u) *_R line\ a\ b\ i + u *_R line\ a\ b\ j = line\ a\ b\ (i + u * (j - i))$
⟨proof⟩

lemma *mult-less-one-less-self*: $0 < x \Longrightarrow i < 1 \Longrightarrow i * x < x$ **for** *i x*::*real*
⟨*proof*⟩

lemma *plus-times-le-one-lemma*: $i + u - i * u \leq 1$ **if** $i \leq 1$ $u \leq 1$ **for** *i u*::*real*
⟨*proof*⟩

lemma *plus-times-less-one-lemma*: $i + u - i * u < 1$ **if** $i < 1$ $u < 1$ **for** *i u*::*real*
⟨*proof*⟩

lemma *line-eq-endpoint-iff* [*simp*]:
line a b i $= b \longleftrightarrow (a = b \vee i = 1)$
$a =$ *line a b i* $\longleftrightarrow (a = b \vee i = 0)$
⟨*proof*⟩

lemma *line-eq-iff* [*simp*]: *line a b x* = *line a b y* $\longleftrightarrow (x = y \vee a = b)$
⟨*proof*⟩

lemma *line-open-segment-iff*:
{*line a b i*<−−<*b*} = *line a b* ‘ {*i*<..<*1*}
if $i < 1$ $a \neq b$
⟨*proof*⟩

lemma *open-segment-line-iff*:
{*a*<−−<*line a b i*} = *line a b* ‘ {*0*<..<*i*}
if $0 < i$ $a \neq b$
⟨*proof*⟩

lemma *line-closed-segment-iff*:
{*line a b i*−−*b*} = *line a b* ‘ {*i*..*1*}
if $i \leq 1$ $a \neq b$
⟨*proof*⟩

lemma *closed-segment-line-iff*:
{*a*−−*line a b i*} = *line a b* ‘ {*0*..*i*}
if $0 < i$ $a \neq b$
⟨*proof*⟩

lemma *closed-segment-line-line-iff*: {*line a b i1*−−*line a b i2*} = *line a b* ‘ {*i1*..*i2*}
if $i1 \leq i2$
⟨*proof*⟩

lemma *line-line1*: *line* (*line a b c*) *b x* = *line a b* $(c + x - c * x)$
⟨*proof*⟩

lemma *line-line2*: *line a* (*line a b c*) *x* = *line a b* $(c*x)$
⟨*proof*⟩

lemma *line-in-subsegment*:

$i1 < 1 \Longrightarrow i2 < 1 \Longrightarrow a \neq b \Longrightarrow line\ a\ b\ i1 \in \{line\ a\ b\ i2{<}{-}{-}{<}b\} \longleftrightarrow i2 < i1$
$\langle proof \rangle$

lemma *line-in-subsegment2*:
$0 < i2 \Longrightarrow 0 < i1 \Longrightarrow a \neq b \Longrightarrow line\ a\ b\ i1 \in \{a{<}{-}{-}{<}line\ a\ b\ i2\} \longleftrightarrow i1 < i2$
$\langle proof \rangle$

lemma *line-in-open-segment-iff* [*simp*]:
$line\ a\ b\ i \in \{a{<}{-}{-}{<}b\} \longleftrightarrow (a \neq b \wedge 0 < i \wedge i < 1)$
$\langle proof \rangle$

1.5 Open Segments

lemma *open-segment-subsegment*:
assumes $x1 \in \{x0{<}{-}{-}{<}x3\}$
$x2 \in \{x1{<}{-}{-}{<}x3\}$
shows $x1 \in \{x0{<}{-}{-}{<}x2\}$
$\langle proof \rangle$

1.6 Syntax

abbreviation *sequentially-at-top*::(*nat*⇒*real*)⇒*bool*
(- $\longrightarrow \infty$) — the is to disambiguate syntax...
where $s \longrightarrow \infty\ \equiv$ *filterlim s at-top sequentially*

abbreviation *sequentially-at-bot*::(*nat*⇒*real*)⇒*bool*
(- $\longrightarrow -\infty$)
where $s \longrightarrow -\infty\ \equiv$ *filterlim s at-bot sequentially*

1.7 Paths

lemma *subpath0-linepath*:
shows $subpath\ 0\ u\ (linepath\ t\ t') = linepath\ t\ (t + u * (t' - t))$
$\langle proof \rangle$

lemma *linepath-image0-right-open-real*:
assumes $t < (t'::real)$
shows $linepath\ t\ t'\ `\ \{0..{<}1\} = \{t..{<}t'\}$
$\langle proof \rangle$

lemma *oriented-subsegment-scale*:
assumes $x1 \in \{a{<}{-}{-}{<}b\}$
assumes $x2 \in \{x1{<}{-}{-}{<}b\}$
obtains e **where** $e > 0\ b{-}a = e *_R (x2{-}x1)$
$\langle proof \rangle$

end

2 Additions to the ODE Library

theory *ODE-Misc*
  imports
    Ordinary-Differential-Equations.ODE-Analysis
    Analysis-Misc
begin

lemma *local-lipschitz-compact-bicomposeE*:
  assumes *ll*: *local-lipschitz T X f*
  assumes *cf*: $\bigwedge x.\ x \in X \Longrightarrow$ *continuous-on I* $(\lambda t.\ f\ t\ x)$
  assumes *cI*: *compact I*
  assumes $I \subseteq T$
  assumes *cv*: *continuous-on I v*
  assumes *cw*: *continuous-on I w*
  assumes *v*: v ‘ $I \subseteq X$
  assumes *w*: w ‘ $I \subseteq X$
  obtains L **where** $L > 0$ $\bigwedge x.\ x \in I \Longrightarrow$ *dist* $(f\ x\ (v\ x))\ (f\ x\ (w\ x)) \leq L *$ *dist* $(v\ x)\ (w\ x)$
$\langle proof \rangle$

2.1 Comparison Principle

lemma *comparison-principle-le*:
  fixes f::*real* $\Rightarrow$ *real* $\Rightarrow$ *real*
    and φ ψ::*real* $\Rightarrow$ *real*
  assumes *ll*: *local-lipschitz X Y f*
  assumes *cf*: $\bigwedge x.\ x \in Y \Longrightarrow$ *continuous-on* $\{a..b\}$ $(\lambda t.\ f\ t\ x)$
  assumes *abX*: $\{a\ ..\ b\} \subseteq X$
  assumes φ': $\bigwedge x.\ x \in \{a\ ..\ b\} \Longrightarrow (\varphi$ *has-real-derivative* $\varphi'\ x)$ $(at\ x)$
  assumes ψ': $\bigwedge x.\ x \in \{a\ ..\ b\} \Longrightarrow (\psi$ *has-real-derivative* $\psi'\ x)$ $(at\ x)$
  assumes φ*-in*: φ ‘ $\{a..b\} \subseteq Y$
  assumes ψ*-in*: ψ ‘ $\{a..b\} \subseteq Y$
  assumes *init*: $\varphi\ a \leq \psi\ a$
  assumes *defect*: $\bigwedge x.\ x \in \{a\ ..\ b\} \Longrightarrow \varphi'\ x - f\ x\ (\varphi\ x) \leq \psi'\ x - f\ x\ (\psi\ x)$
  shows $\forall\, x \in \{a\ ..\ b\}.\ \varphi\ x \leq \psi\ x$ (**is** *?th1*)

  $\langle proof \rangle$

lemma *local-lipschitz-mult*:
  shows *local-lipschitz* (*UNIV*::*real set*) (*UNIV*::*real set*) $(*)$
  $\langle proof \rangle$

lemma *comparison-principle-le-linear*:
  fixes φ :: *real* $\Rightarrow$ *real*
  assumes *continuous-on* $\{a..b\}$ g
  assumes $(\bigwedge t.\ t \in \{a..b\} \Longrightarrow (\varphi$ *has-real-derivative* $\varphi'\ t)$ $(at\ t))$
  assumes $\varphi\ a \leq 0$
  assumes $(\bigwedge t.\ t \in \{a..b\} \Longrightarrow \varphi'\ t \leq g\ t *_R \varphi\ t)$
  shows $\forall\, t \in \{a..b\}.\ \varphi\ t \leq 0$

⟨*proof*⟩

2.2 Locally Lipschitz ODEs

context *ll-on-open-it* **begin**

lemma *flow-lipschitzE*:
  assumes $\{a .. b\} \subseteq$ *existence-ivl t0 x*
  obtains L **where** L*−lipschitz-on* $\{a .. b\}$ (*flow t0 x*)
⟨*proof*⟩

lemma *flow-undefined0*: $t \notin$ *existence-ivl t0 x* $\Longrightarrow$ *flow t0 x t* $=$ *0*
  ⟨*proof*⟩

lemma *csols-undefined*: $x \notin X \Longrightarrow$ *csols t0 x* $= \{\}$
  ⟨*proof*⟩

lemmas *existence-ivl-undefined* = *existence-ivl-empty2*

end

2.3 Reverse flow as Sublocale

lemma *range-preflect-0*[*simp*]: *range* (*preflect 0*) = *UNIV*
  ⟨*proof*⟩
lemma *range-uminus*[*simp*]: *range uminus* = (*UNIV*::′*a*::*ab-group-add set*)
  ⟨*proof*⟩

context *auto-ll-on-open* **begin**

sublocale *rev*: *auto-ll-on-open* $-f$ **rewrites** $-(-f) = f$
   ⟨*proof*⟩

lemma *existence-ivl-eq-rev0*: *existence-ivl0 y* = *uminus* ‘ *rev.existence-ivl0 y* **for** *y*
  ⟨*proof*⟩

lemma *rev-existence-ivl-eq0*: *rev.existence-ivl0 y* = *uminus* ‘ *existence-ivl0 y* **for** *y*
  ⟨*proof*⟩

lemma *flow-eq-rev0*: *flow0 y t* = *rev.flow0 y* $(-t)$ **for** *y t*
  ⟨*proof*⟩

lemma *rev-eq-flow*: *rev.flow0 y t* = *flow0 y* $(-t)$ **for** *y t*
  ⟨*proof*⟩

lemma *rev-flow-image-eq*: *rev.flow0 x* ‘ *S* = *flow0 x* ‘ (*uminus* ‘ *S*)
  ⟨*proof*⟩

lemma *flow-image-eq-rev*: *flow0 x* ‘ *S* = *rev.flow0 x* ‘ (*uminus* ‘ *S*)
  ⟨*proof*⟩

end

context *c1-on-open* **begin**

sublocale *rev*: *c1-on-open* $-f$ $-f'$ **rewrites** $-(-f) = f$ **and** $-(-f') = f'$
⟨*proof*⟩

end

context *c1-on-open-euclidean* **begin**

sublocale *rev*: *c1-on-open-euclidean* $-f$ $-f'$ **rewrites** $-(-f) = f$ **and** $-(-f') = f'$
⟨*proof*⟩

end

2.4 Autonomous LL ODE : Existence Interval and trapping on the interval

lemma *bdd-above-is-intervalI*: *bdd-above I*
if *is-interval I* $a \le b$ $a \in I$ $b \notin I$ **for** *I*::*real set*
⟨*proof*⟩

lemma *bdd-below-is-intervalI*: *bdd-below I*
if *is-interval I* $a \le b$ $a \notin I$ $b \in I$ **for** *I*::*real set*
⟨*proof*⟩

context *auto-ll-on-open* **begin**

lemma *open-existence-ivl0*:
assumes $x : x \in X$
shows $\exists\, a\ b.\ a < 0 \wedge 0 < b \wedge \{a..b\} \subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *open-existence-ivl'*:
assumes $x : x \in X$
obtains a **where** $a > 0$ $\{-a..a\} \subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *open-existence-ivl-on-compact*:
assumes C: $C \subseteq X$ **and** *compact C* $C \neq \{\}$
obtains a **where** $a > 0$ $\bigwedge x.\ x \in C \Longrightarrow \{-a..a\} \subseteq$ *existence-ivl0 x*
⟨*proof*⟩

definition *trapped-forward x K* $\longleftrightarrow$ (*flow0 x* ‘ (*existence-ivl0 x* $\cap$ {*0*..}) $\subseteq K$)
— TODO: use this for backwards trapped, invariant, and all assumptions

definition *trapped-backward x K* $\longleftrightarrow$ (*flow0 x* ‘ (*existence-ivl0 x* $\cap$ {..*0*}) $\subseteq$ *K*)

definition *trapped x K* $\longleftrightarrow$ *trapped-forward x K* $\wedge$ *trapped-backward x K*

lemma *trapped-iff-on-existence-ivl0*:
  trapped x K $\longleftrightarrow$ (*flow0 x* ‘ (*existence-ivl0 x*) $\subseteq$ *K*)
  ⟨*proof*⟩
end

context *auto-ll-on-open* **begin**

lemma *infinite-rev-existence-ivl0-rewrites*:
  {*0*..} $\subseteq$ *rev.existence-ivl0 x* $\longleftrightarrow$ {..*0*} $\subseteq$ *existence-ivl0 x*
  {..*0*} $\subseteq$ *rev.existence-ivl0 x* $\longleftrightarrow$ {*0*..} $\subseteq$ *existence-ivl0 x*
  ⟨*proof*⟩

lemma *trapped-backward-iff-rev-trapped-forward*:
  trapped-backward x K $\longleftrightarrow$ *rev.trapped-forward x K*
  ⟨*proof*⟩

If solution is trapped in a compact set at some time on its existence interval then it is trapped forever

lemma *trapped-sol-right*:
  — TODO: when building on afp-devel (??? outdated): https://bitbucket.org/isa-afp/afp-devel/commits/0c3edf9248d5389197f248c723b625c419e4d3eb
  assumes *compact K K* $\subseteq$ *X*
  assumes *x* $\in$ *X trapped-forward x K*
  shows {*0*..} $\subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *trapped-sol-right-gen*:
  assumes *compact K K* $\subseteq$ *X*
  assumes *t* $\in$ *existence-ivl0 x trapped-forward* (*flow0 x t*) *K*
  shows {*t*..} $\subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *trapped-sol-left*:
  — TODO: when building on afp-devel: https://bitbucket.org/isa-afp/afp-devel/commits/0c3edf9248d5389197f248c723b625c419e4d3eb
  assumes *compact K K* $\subseteq$ *X*
  assumes *x* $\in$ *X trapped-backward x K*
  shows {..*0*} $\subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *trapped-sol-left-gen*:
  assumes *compact K K* $\subseteq$ *X*
  assumes *t* $\in$ *existence-ivl0 x trapped-backward* (*flow0 x t*) *K*
  shows {..*t*} $\subseteq$ *existence-ivl0 x*
⟨*proof*⟩

lemma *trapped-sol*:
  assumes $compact\ K\ K \subseteq X$
  assumes $x \in X\ trapped\ x\ K$
  shows $existence\text{-}ivl0\ x = UNIV$
  ⟨*proof*⟩

lemma *regular-locally-noteq*:— TODO: should be true in *ll-on-open-it*
  assumes $x \in X\ f\ x \neq 0$
  shows $eventually\ (\lambda t.\ flow0\ x\ t \neq x)\ (at\ 0)$
⟨*proof*⟩

lemma *compact-max-time-flow-in-closed*:
  assumes *closed M* **and** *t-ex*: $t \in existence\text{-}ivl0\ x$
  shows $compact\ \{s \in \{0..t\}.\ flow0\ x\ `\ \{0..s\} \subseteq M\}$ (**is** *compact ?C*)
  ⟨*proof*⟩

lemma *flow-in-closed-max-timeE*:
  assumes $closed\ M\ t \in existence\text{-}ivl0\ x\ 0 \leq t\ x \in M$
  obtains T **where** $0 \leq T\ T \leq t\ flow0\ x\ `\ \{0..T\} \subseteq M$
    $\bigwedge s'.\ 0 \leq s' \Longrightarrow s' \leq t \Longrightarrow flow0\ x\ `\ \{0..s'\} \subseteq M \Longrightarrow s' \leq T$
⟨*proof*⟩

lemma *flow-leaves-closed-at-frontierE*:
  assumes *closed M* **and** *t-ex*: $t \in existence\text{-}ivl0\ x$ **and** $0 \leq t\ x \in M\ flow0\ x\ t \notin M$
  obtains s **where** $0 \leq s\ s < t\ flow0\ x\ `\ \{0..s\} \subseteq M$
    $flow0\ x\ s \in frontier\ M$
    $\exists_F\ s'\ in\ at\text{-}right\ s.\ flow0\ x\ s' \notin M$
⟨*proof*⟩

2.5 Connectedness

lemma *fcontX*:
  shows *continuous-on X f*
  ⟨*proof*⟩

lemma *fcontx*:
  assumes $x \in X$
  shows $continuous\ (at\ x)\ f$
⟨*proof*⟩

lemma *continuous-at-imp-cball*:
  assumes $continuous\ (at\ x)\ g$
  assumes $g\ x > (0::real)$
  obtains r **where** $r > 0\ \forall y \in cball\ x\ r.\ g\ y > 0$
⟨*proof*⟩

flow0 is *path-connected*

lemma *flow0-path-connected-time*:
  assumes *ts* ⊆ *existence-ivl0 x path-connected ts*
  shows *path-connected* (*flow0 x* ‘ *ts*)
⟨*proof*⟩

lemma *flow0-path-connected*:
  assumes *path-connected D*
    path-connected ts
    ⋀*x*. *x* ∈ *D* ⟹ *ts* ⊆ *existence-ivl0 x*
  shows *path-connected* ((λ(*x*, *y*). *flow0 x y*) ‘ (*D* × *ts*))
⟨*proof*⟩

end

2.6 Return Time and Implicit Function Theorem

context *c1-on-open-euclidean* **begin**

lemma *flow-implicit-function*:
  — TODO: generalization of ⟦*returns-to* {*x* ∈ *?S*. *?s x* = *0*} *?x*; *closed ?S*; ⋀*x*. (*?s has-derivative blinfun-apply* (*?Ds x*)) (*at x*); *isCont ?Ds* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*); *blinfun-apply* (*?Ds* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*)) (*f* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*)) ≠ *0*; ⋀*u e*. ⟦*?s* (*flow0 ?x* (*u ?x*)) = *0*; *u ?x* = *return-time* {*x* ∈ *?S*. *?s x* = *0*} *?x*; ⋀*y*. *y* ∈ *cball ?x e* ⟹ *?s* (*flow0 y* (*u y*)) = *0*; *continuous-on* (*cball ?x e*) *u*; (λ*t*. (*t*, *u t*)) ‘ *cball ?x e* ⊆ *Sigma X existence-ivl0*; *0* < *e*; (*u has-derivative blinfun-apply* (− *blinfun-scaleR-left* (*inverse* (*blinfun-apply* (*?Ds* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*)) (*f* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*)))) o_L (*?Ds* (*poincare-map* {*x* ∈ *?S*. *?s x* = *0*} *?x*) o_L *flowderiv ?x* (*return-time* {*x* ∈ *?S*. *?s x* = *0*} *?x*)) o_L *embed1-blinfun*)) (*at ?x*)⟧ ⟹ *?thesis*⟧ ⟹ *?thesis*!
  fixes *s*::′*a*::*euclidean-space* ⇒ *real* **and** *S*::′*a set*
  assumes *t*: *t* ∈ *existence-ivl0 x* **and** *x*: *x* ∈ *X* **and** *st*: *s* (*flow0 x t*) = *0*
  assumes *Ds*: ⋀*x*. (*s has-derivative blinfun-apply* (*Ds x*)) (*at x*)
  assumes *DsC*: *isCont Ds* (*flow0 x t*)
  assumes *nz*: *Ds* (*flow0 x t*) (*f* (*flow0 x t*)) ≠ *0*
  obtains *u e*
  where *s* (*flow0 x* (*u x*)) = *0*
    u x = *t*
    (⋀*y*. *y* ∈ *cball x e* ⟹ *s* (*flow0 y* (*u y*)) = *0*)
    continuous-on (*cball x e*) *u*
    (λ*t*. (*t*, *u t*)) ‘ *cball x e* ⊆ *Sigma X existence-ivl0*
    0 < *e* (*u has-derivative* (− *blinfun-scaleR-left*
          (*inverse* (*blinfun-apply* (*Ds* (*flow0 x t*)) (*f* (*flow0 x t*)))) o_L
            (*Ds* (*flow0 x t*) o_L *flowderiv x t*) o_L *embed1-blinfun*)) (*at x*)
⟨*proof*⟩

lemma *flow-implicit-function-at*:
  fixes *s*::′*a*::*euclidean-space* ⇒ *real* **and** *S*::′*a set*
  assumes *x*: *x* ∈ *X* **and** *st*: *s x* = *0*
  assumes *Ds*: ⋀*x*. (*s has-derivative blinfun-apply* (*Ds x*)) (*at x*)

assumes *DsC*: $isCont\ Ds\ x$
assumes *nz*: $Ds\ x\ (f\ x) \neq 0$
assumes *pos*: $e > 0$
obtains $u\ d$
where
$0 < d$
$u\ x = 0$
$\bigwedge y.\ y \in cball\ x\ d \Longrightarrow s\ (flow0\ y\ (u\ y)) = 0$
$\bigwedge y.\ y \in cball\ x\ d \Longrightarrow |u\ y| < e$
$\bigwedge y.\ y \in cball\ x\ d \Longrightarrow u\ y \in existence\text{-}ivl0\ y$
$continuous\text{-}on\ (cball\ x\ d)\ u$
$(u\ has\text{-}derivative\ -Ds\ x\ /_R\ (Ds\ x)\ (f\ x))\ (at\ x)$
$\langle proof \rangle$

lemma *returns-to-implicit-function-gen*:
— TODO: generalizes proof of $[\![returns\text{-}to\ \{x \in ?S.\ ?s\ x = 0\}\ ?x;\ closed\ ?S;\ \bigwedge x.\ (?s\ has\text{-}derivative\ blinfun\text{-}apply\ (?Ds\ x))\ (at\ x);\ isCont\ ?Ds\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x);\ blinfun\text{-}apply\ (?Ds\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x))\ (f\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x)) \neq 0;\ \bigwedge u\ e.\ [\![?s\ (flow0\ ?x\ (u\ ?x)) = 0;\ u\ ?x = return\text{-}time\ \{x \in ?S.\ ?s\ x = 0\}\ ?x;\ \bigwedge y.\ y \in cball\ ?x\ e \Longrightarrow ?s\ (flow0\ y\ (u\ y)) = 0;\ continuous\text{-}on\ (cball\ ?x\ e)\ u;\ (\lambda t.\ (t,\ u\ t))\ `\ cball\ ?x\ e \subseteq Sigma\ X\ existence\text{-}ivl0;\ 0 < e;\ (u\ has\text{-}derivative\ blinfun\text{-}apply\ (-\ blinfun\text{-}scaleR\text{-}left\ (inverse\ (blinfun\text{-}apply\ (?Ds\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x))\ (f\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x))))\ o_L\ (?Ds\ (poincare\text{-}map\ \{x \in ?S.\ ?s\ x = 0\}\ ?x)\ o_L\ flowderiv\ ?x\ (return\text{-}time\ \{x \in ?S.\ ?s\ x = 0\}\ ?x))\ o_L\ embed1\text{-}blinfun))\ (at\ ?x)]\!] \Longrightarrow ?thesis]\!] \Longrightarrow ?thesis$!
fixes $s::'a::euclidean\text{-}space \Rightarrow real$
assumes *rt*: $returns\text{-}to\ \{x \in S.\ s\ x = 0\}\ x$ (**is** $returns\text{-}to\ ?P\ x$)
assumes *cS*: $closed\ S$
assumes *Ds*: $\bigwedge x.\ (s\ has\text{-}derivative\ blinfun\text{-}apply\ (Ds\ x))\ (at\ x)$
$isCont\ Ds\ (poincare\text{-}map\ ?P\ x)$
$Ds\ (poincare\text{-}map\ ?P\ x)\ (f\ (poincare\text{-}map\ ?P\ x)) \neq 0$
obtains $u\ e$
where $s\ (flow0\ x\ (u\ x)) = 0$
$u\ x = return\text{-}time\ ?P\ x$
$(\bigwedge y.\ y \in cball\ x\ e \Longrightarrow s\ (flow0\ y\ (u\ y)) = 0)$
$continuous\text{-}on\ (cball\ x\ e)\ u$
$(\lambda t.\ (t,\ u\ t))\ `\ cball\ x\ e \subseteq Sigma\ X\ existence\text{-}ivl0$
$0 < e\ (u\ has\text{-}derivative\ (-\ blinfun\text{-}scaleR\text{-}left$
$(inverse\ (blinfun\text{-}apply\ (Ds\ (poincare\text{-}map\ ?P\ x))\ (f\ (poincare\text{-}map\ ?P\ x))))\ o_L$
$(Ds\ (poincare\text{-}map\ ?P\ x)\ o_L\ flowderiv\ x\ (return\text{-}time\ ?P\ x))\ o_L\ embed1\text{-}blinfun))\ (at\ x)$
$\langle proof \rangle$

c.f. Perko Section 3.7 Lemma 2 part 1.

lemma *flow-transversal-surface-finite-intersections*:
fixes $s::'a \Rightarrow 'b::real\text{-}normed\text{-}vector$
and $Ds::'a \Rightarrow 'a \Rightarrow_L 'b$
assumes $closed\ S$

assumes $\bigwedge x.$ (s *has-derivative* (Ds x)) (*at* x)
assumes $\bigwedge x.$ $x \in S \Longrightarrow s\ x = 0 \Longrightarrow Ds\ x\ (f\ x) \neq 0$
assumes $a \leq b$ $\{a\ ..\ b\} \subseteq$ *existence-ivl0* x
shows *finite* $\{t{\in}\{a..b\}.$ *flow0* $x\ t \in \{x \in S.\ s\ x = 0\}\}$
— TODO: define notion of (compact/closed)-(continuous/differentiable/C1)-surface?
⟨*proof*⟩

lemma *uniform-limit-flow0-state*:— TODO: is that something more general?
assumes *compact* C
assumes $C \subseteq X$
shows *uniform-limit* C ($\lambda s\ x.$ *flow0* $x\ s$) ($\lambda x.$ *flow0* $x\ 0$) (*at* 0)
⟨*proof*⟩

end

2.7 Fixpoints

context *auto-ll-on-open* **begin**

lemma *fixpoint-sol*:
assumes $x \in X$ $f\ x = 0$
shows *existence-ivl0* x = *UNIV* *flow0* $x\ t = x$
⟨*proof*⟩

end

end

3 Invariance

theory *Invariance*
imports *ODE-Misc*
begin

context *auto-ll-on-open* **begin**

definition *invariant* $M \longleftrightarrow (\forall\, x{\in}M.$ *trapped* $x\ M)$

definition *positively-invariant* $M \longleftrightarrow (\forall\, x{\in}M.$ *trapped-forward* $x\ M)$

definition *negatively-invariant* $M \longleftrightarrow (\forall\, x{\in}M.$ *trapped-backward* $x\ M)$

lemma *positively-invariant-iff*:
positively-invariant $M \longleftrightarrow$
$(\bigcup x{\in}M.$ *flow0* x ‘ (*existence-ivl0* $x \cap \{0..\})) \subseteq M$
⟨*proof*⟩

lemma *negatively-invariant-iff*:

negatively-invariant $M \longleftrightarrow$
$(\bigcup x{\in}M.\ flow0\ x\ `\ (existence\text{-}ivl0\ x \cap \{..0\})) \subseteq M$
⟨*proof*⟩

lemma *invariant-iff-pos-and-neg-invariant*:
invariant $M \longleftrightarrow$ *positively-invariant* $M \wedge$ *negatively-invariant* M
⟨*proof*⟩

lemma *invariant-iff*:
invariant $M \longleftrightarrow (\bigcup x{\in}M.\ flow0\ x\ `\ (existence\text{-}ivl0\ x)) \subseteq M$
⟨*proof*⟩

lemma *positively-invariant-restrict-dom*: *positively-invariant* $M =$ *positively-invariant* $(M \cap X)$
⟨*proof*⟩

lemma *negatively-invariant-restrict-dom*: *negatively-invariant* $M =$ *negatively-invariant* $(M \cap X)$
⟨*proof*⟩

lemma *invariant-restrict-dom*: *invariant* $M =$ *invariant* $(M \cap X)$
⟨*proof*⟩

end context *auto-ll-on-open* **begin**

lemma *positively-invariant-eq-rev*: *positively-invariant* $M =$ *rev.negatively-invariant* M
⟨*proof*⟩

lemma *negatively-invariant-eq-rev*: *negatively-invariant* $M =$ *rev.positively-invariant* M
⟨*proof*⟩

lemma *invariant-eq-rev*: *invariant* $M =$ *rev.invariant* M
⟨*proof*⟩

lemma *negatively-invariant-complI*: *negatively-invariant* $(X-M)$ **if** *positively-invariant* M
⟨*proof*⟩

end context *auto-ll-on-open* **begin**

lemma *negatively-invariant-complD*: *positively-invariant* M **if** *negatively-invariant* $(X-M)$
⟨*proof*⟩

lemma *pos-invariant-iff-compl-neg-invariant*: *positively-invariant* $M \longleftrightarrow$ *negatively-invariant* $(X - M)$

⟨*proof*⟩

lemma *neg-invariant-iff-compl-pos-invariant*:
shows *negatively-invariant* $M \longleftrightarrow$ *positively-invariant* $(X - M)$
⟨*proof*⟩

lemma *invariant-iff-compl-invariant*:
shows *invariant* $M \longleftrightarrow$ *invariant* $(X - M)$
⟨*proof*⟩

lemma *invariant-iff-pos-invariant-and-compl-pos-invariant*:
shows *invariant* $M \longleftrightarrow$ *positively-invariant* $M \wedge$ *positively-invariant* $(X{-}M)$
⟨*proof*⟩

end

3.1 Tools for proving invariance

context *auto-ll-on-open* **begin**

lemma *positively-invariant-left-inter*:
assumes *positively-invariant* C
assumes $\forall x \in C \cap D.$ *trapped-forward* x D
shows *positively-invariant* $(C \cap D)$
⟨*proof*⟩

lemma *trapped-forward-le*:
fixes $V :: 'a \Rightarrow real$
assumes $V\ x \leq 0$
assumes *contg*: *continuous-on* (*flow0* x ‘ (*existence-ivl0* $x \cap \{0..\}$)) g
assumes $\bigwedge x.$ (V *has-derivative* $V'\ x$) (*at* x)
assumes $\bigwedge s.\ s \in$ *existence-ivl0* $x \cap \{0..\} \Longrightarrow V'$ (*flow0* x s) (f (*flow0* x s)) $\leq g$ (*flow0* x s) $*$ V (*flow0* x s)
shows *trapped-forward* x $\{x.\ V\ x \leq 0\}$
⟨*proof*⟩

lemma *positively-invariant-le-domain*:
fixes $V :: 'a \Rightarrow real$
assumes *positively-invariant* D
assumes *contg*: *continuous-on* D g
assumes $\bigwedge x.$ (V *has-derivative* $V'\ x$) (*at* x)
assumes $\bigwedge s.\ s \in D \Longrightarrow V'\ s\ (f\ s) \leq g\ s * V\ s$
shows *positively-invariant* $(D \cap \{x.\ V\ x \leq 0\})$
⟨*proof*⟩

lemma *positively-invariant-barrier-domain*:
fixes $V :: 'a \Rightarrow real$
assumes *positively-invariant* D
assumes $\bigwedge x.$ (V *has-derivative* $V'\ x$) (*at* x)

assumes *continuous-on* D $(\lambda x.\ V'\ x\ (f\ x))$
assumes $\bigwedge s.\ s \in D \Longrightarrow V\ s = 0 \Longrightarrow V'\ s\ (f\ s) < 0$
shows *positively-invariant* $(D \cap \{x.\ V\ x \leq 0\})$
⟨*proof*⟩

lemma *positively-invariant-UNIV*:
shows *positively-invariant UNIV*
⟨*proof*⟩

lemma *positively-invariant-conj*:
assumes *positively-invariant* C
assumes *positively-invariant* D
shows *positively-invariant* $(C \cap D)$
⟨*proof*⟩

lemma *positively-invariant-le*:
fixes $V :: 'a \Rightarrow real$
assumes *contg*: *continuous-on UNIV* g
assumes $\bigwedge x.\ (V\ \textit{has-derivative}\ V'\ x)\ (at\ x)$
assumes $\bigwedge s.\ V'\ s\ (f\ s) \leq g\ s * V\ s$
shows *positively-invariant* $\{x.\ V\ x \leq 0\}$
⟨*proof*⟩

lemma *positively-invariant-barrier*:
fixes $V :: 'a \Rightarrow real$
assumes $\bigwedge x.\ (V\ \textit{has-derivative}\ V'\ x)\ (at\ x)$
assumes *continuous-on UNIV* $(\lambda x.\ V'\ x\ (f\ x))$
assumes $\bigwedge s.\ V\ s = 0 \Longrightarrow V'\ s\ (f\ s) < 0$
shows *positively-invariant* $\{x.\ V\ x \leq 0\}$
⟨*proof*⟩

end

end

4 Limit Sets

theory *Limit-Set*
imports *Invariance*
begin

context *auto-ll-on-open* **begin**

Positive limit point, assuming $\{0..\} \subseteq$ *existence-ivl0* x

definition ω*-limit-point* $x\ p \longleftrightarrow$
$\{0..\} \subseteq$ *existence-ivl0* $x\ \wedge$
$(\exists s.\ s \longrightarrow \infty \wedge (\textit{flow0}\ x \circ s) \longrightarrow p)$

Also called the ω-limit set of x

definition ω*-limit-set* $x = \{p.\ \omega$*-limit-point* $x\ p\}$

definition α*-limit-point* $x\ p \longleftrightarrow$
$\{..0\} \subseteq$ *existence-ivl0* $x\ \wedge$
$(\exists s.\ s \longrightarrow -\infty \wedge (flow0\ x \circ s) \longrightarrow p)$

Also called the α-limit set of x

definition α*-limit-set* $x =$
$\{p.\ \alpha$*-limit-point* $x\ p\}$

end context *auto-ll-on-open* **begin**

lemma α*-limit-point-eq-rev*: α*-limit-point* $x\ p$ = *rev.*ω*-limit-point* $x\ p$
⟨*proof*⟩

lemma α*-limit-set-eq-rev*: α*-limit-set* x = *rev.*ω*-limit-set* x
⟨*proof*⟩

lemma ω*-limit-pointE*:
assumes ω*-limit-point* $x\ p$
obtains s **where**
filterlim s at-top sequentially
$(flow0\ x \circ s) \longrightarrow p$
$\forall n.\ b \leq s\ n$
⟨*proof*⟩

lemma ω*-limit-set-eq*:
assumes $\{0..\} \subseteq$ *existence-ivl0* x
shows ω*-limit-set* $x = (INF\ \tau \in \{0..\}.\ closure\ (flow0\ x\ `\ \{\tau..\}))$
⟨*proof*⟩

lemma ω*-limit-set-empty*:
assumes $\neg$ ($\{0..\} \subseteq$ *existence-ivl0* x)
shows ω*-limit-set* $x = \{\}$
⟨*proof*⟩

lemma ω*-limit-set-closed*: *closed* (ω*-limit-set* x)
⟨*proof*⟩

lemma ω*-limit-set-positively-invariant*:
shows *positively-invariant* (ω*-limit-set* x)
⟨*proof*⟩

lemma ω*-limit-set-invariant*:
shows *invariant* (ω*-limit-set* x)
⟨*proof*⟩

end context *auto-ll-on-open* **begin**

lemma *α-limit-set-eq*:
  assumes $\{..0\} \subseteq$ *existence-ivl0 x*
  shows *α-limit-set x* $=$ (*INF* $\tau \in \{..0\}$. *closure* (*flow0 x* ‘ $\{..\tau\}$))
  ⟨*proof*⟩

lemma *α-limit-set-closed*:
  shows *closed* (*α-limit-set x*)
  ⟨*proof*⟩

lemma *α-limit-set-positively-invariant*:
  shows *negatively-invariant* (*α-limit-set x*)
  ⟨*proof*⟩

lemma *α-limit-set-invariant*:
  shows *invariant* (*α-limit-set x*)
  ⟨*proof*⟩

Fundamental properties of the positive limit set

context
  fixes *x K*
  assumes *K*: *compact K K* $\subseteq$ *X*
  assumes *x*: *x* $\in$ *X trapped-forward x K*
begin

Bunch of facts for what's to come

private lemma *props*:
  shows $\{0..\} \subseteq$ *existence-ivl0 x seq-compact K*
  ⟨*proof*⟩ **lemma** *flowimg*:
  shows *flow0 x* ‘ (*existence-ivl0 x* $\cap$ $\{0..\}$) $=$ *flow0 x* ‘ $\{0..\}$
  ⟨*proof*⟩

lemma *ω-limit-set-in-compact-subset*:
  shows *ω-limit-set x* $\subseteq$ *K*
  ⟨*proof*⟩

lemma *ω-limit-set-in-compact-compact*:
  shows *compact* (*ω-limit-set x*)
⟨*proof*⟩

lemma *ω-limit-set-in-compact-nonempty*:
  shows *ω-limit-set x* $\neq$ $\{\}$
⟨*proof*⟩

lemma *ω-limit-set-in-compact-existence*:
  shows $\bigwedge y$. $y \in$ *ω-limit-set x* $\Longrightarrow$ *existence-ivl0 y* $=$ *UNIV*
⟨*proof*⟩

lemma *ω-limit-set-in-compact-tendsto*:

shows $((\lambda t.\ \mathit{infdist}\ (\mathit{flow0}\ x\ t)\ (\omega\text{-}\mathit{limit}\text{-}\mathit{set}\ x)) \longrightarrow 0)\ \mathit{at}\text{-}\mathit{top}$
⟨*proof*⟩

lemma ω*-limit-set-in-compact-connected*:
shows *connected* (ω*-limit-set* x)
⟨*proof*⟩

lemma ω*-limit-set-in-compact-*ω*-limit-set-contained*:
shows $\forall\, y \in \omega\text{-}\mathit{limit}\text{-}\mathit{set}\ x.\ \omega\text{-}\mathit{limit}\text{-}\mathit{set}\ y \subseteq \omega\text{-}\mathit{limit}\text{-}\mathit{set}\ x$
⟨*proof*⟩

lemma ω*-limit-set-in-compact-*α*-limit-set-contained*:
assumes *zpx*: $z \in \omega\text{-}\mathit{limit}\text{-}\mathit{set}\ x$
shows $\alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ z \subseteq \omega\text{-}\mathit{limit}\text{-}\mathit{set}\ x$
⟨*proof*⟩

end

Fundamental properties of the negative limit set

end context *auto-ll-on-open* **begin**

context
fixes x K
assumes x: $x \in X$ *trapped-backward* x K
assumes K: *compact* K $K \subseteq X$
begin

private lemma *xrev*: $x \in X$ *rev.trapped-forward* x K
⟨*proof*⟩

lemma α*-limit-set-in-compact-subset*: $\alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x \subseteq K$
and α*-limit-set-in-compact-compact*: *compact* (α*-limit-set* x)
and α*-limit-set-in-compact-nonempty*: $\alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x \neq \{\}$
and α*-limit-set-in-compact-connected*: *connected* (α*-limit-set* x)
and α*-limit-set-in-compact-*α*-limit-set-contained*:
$\forall\, y \in \alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x.\ \alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ y \subseteq \alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x$
and α*-limit-set-in-compact-tendsto*: $((\lambda t.\ \mathit{infdist}\ (\mathit{flow0}\ x\ t)\ (\alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x)) \longrightarrow 0)\ \mathit{at}\text{-}\mathit{bot}$
⟨*proof*⟩

lemma α*-limit-set-in-compact-existence*:
shows $\bigwedge y.\ y \in \alpha\text{-}\mathit{limit}\text{-}\mathit{set}\ x \Longrightarrow \mathit{existence}\text{-}\mathit{ivl0}\ y = \mathit{UNIV}$
⟨*proof*⟩

end
end

end

5 Periodic Orbits

theory *Periodic-Orbit*
  imports
    Ordinary-Differential-Equations.ODE-Analysis
    Analysis-Misc
    ODE-Misc
    Limit-Set
begin

Definition of closed and periodic orbits and their associated properties

context *auto-ll-on-open*
begin

TODO: not sure if the "closed orbit" terminology is standard Closed orbits have some non-zero recurrence time T where the flow returns to the initial state The period of a closed orbit is the infimum of all positive recurrence times Periodic orbits are the subset of closed orbits where the period is non-zero

definition *closed-orbit* $x \longleftrightarrow$
  $(\exists\, T \in \textit{existence-ivl0}\ x.\ T \neq 0 \wedge \textit{flow0}\ x\ T = x)$

definition *period* $x =$
  $\textit{Inf}\ \{T \in \textit{existence-ivl0}\ x.\ T > 0 \wedge \textit{flow0}\ x\ T = x\}$

definition *periodic-orbit* $x \longleftrightarrow$
  $\textit{closed-orbit}\ x \wedge \textit{period}\ x > 0$

lemma *recurrence-time-flip-sign*:
  assumes $T \in \textit{existence-ivl0}\ x\ \textit{flow0}\ x\ T = x$
  shows $-T \in \textit{existence-ivl0}\ x\ \textit{flow0}\ x\ (-T) = x$
  ⟨*proof*⟩

lemma *closed-orbit-recurrence-times-nonempty*:
  assumes *closed-orbit* x
  shows $\{T \in \textit{existence-ivl0}\ x.\ T > 0 \wedge \textit{flow0}\ x\ T = x\} \neq \{\}$
  ⟨*proof*⟩

lemma *closed-orbit-recurrence-times-bdd-below*:
  shows $\textit{bdd-below}\ \{T \in \textit{existence-ivl0}\ x.\ T > 0 \wedge \textit{flow0}\ x\ T = x\}$
  ⟨*proof*⟩

lemma *closed-orbit-period-nonneg*:
  assumes *closed-orbit* x
  shows *period* $x \geq 0$
  ⟨*proof*⟩

lemma *closed-orbit-in-domain*:
  assumes *closed-orbit* x

shows $x \in X$
⟨*proof*⟩

lemma *closed-orbit-global-existence*:
assumes *closed-orbit x*
shows *existence-ivl0 x* $=$ *UNIV*
⟨*proof*⟩

lemma *recurrence-time-multiples*:
fixes n::*nat*
assumes $T \in$ *existence-ivl0 x* $T \neq 0$ *flow0 x T* $= x$
shows $\bigwedge t$. *flow0 x* $(t+T*n) =$ *flow0 x t*
⟨*proof*⟩

lemma *nasty-arithmetic1*:
fixes t T::*real*
assumes $T > 0$ $t \geq 0$
obtains q r **where** $t = (q$::*nat*$) * T + r$ $0 \leq r$ $r < T$
⟨*proof*⟩

lemma *nasty-arithmetic2*:
fixes t T::*real*
assumes $T > 0$ $t \leq 0$
obtains q r **where** $t = (q$::*nat*$) * (-T) + r$ $0 \leq r$ $r < T$
⟨*proof*⟩

lemma *recurrence-time-restricts-compact-flow*:
assumes $T \in$ *existence-ivl0 x* $T > 0$ *flow0 x T* $= x$
shows *flow0 x* ‘ *UNIV* $=$ *flow0 x* ‘ $\{0..T\}$
⟨*proof*⟩

lemma *closed-orbitI*:
assumes $t \neq t'$ $t \in$ *existence-ivl0 y* $t' \in$ *existence-ivl0 y*
assumes *flow0 y t* $=$ *flow0 y* t'
shows *closed-orbit y*
⟨*proof*⟩

lemma *flow0-image-UNIV*:
assumes *existence-ivl0 x* $=$ *UNIV*
shows *flow0* (*flow0 x t*) ‘ S $=$ *flow0 x* ‘ $(\lambda s.\ s + t)$ ‘ S
⟨*proof*⟩

lemma *recurrence-time-restricts-compact-flow'*:
assumes $t < t'$ $t \in$ *existence-ivl0 y* $t' \in$ *existence-ivl0 y*
assumes *flow0 y t* $=$ *flow0 y* t'
shows *flow0 y* ‘ *UNIV* $=$ *flow0 y* ‘ $\{t..t'\}$
⟨*proof*⟩

lemma *closed-orbitE'*:
assumes *closed-orbit x*
obtains T **where** $T > 0$ $\bigwedge t$ $(n{::}nat)$. *flow0* x $(t+T*n) =$ *flow0* x t
⟨*proof*⟩

lemma *closed-orbitE*:
assumes *closed-orbit x*
obtains T **where** $T > 0$ $\bigwedge t$. *flow0* x $(t+T) =$ *flow0* x t
⟨*proof*⟩

lemma *closed-orbit-flow-compact*:
assumes *closed-orbit x*
shows *compact*(*flow0* x ‘ *UNIV*)
⟨*proof*⟩

lemma *fixed-point-imp-closed-orbit-period-zero*:
assumes $x \in X$
assumes $f\ x = 0$
shows *closed-orbit x period* $x = 0$
⟨*proof*⟩

lemma *closed-orbit-period-zero-fixed-point*:
assumes *closed-orbit x period* $x = 0$
shows $f\ x = 0$
⟨*proof*⟩

lemma *closed-orbit-subset-ω-limit-set*:
assumes *closed-orbit x*
shows *flow0* x ‘ *UNIV* $\subseteq$ *ω-limit-set x*
⟨*proof*⟩

lemma *closed-orbit-ω-limit-set*:
assumes *closed-orbit x*
shows *flow0* x ‘ *UNIV* $=$ *ω-limit-set x*
⟨*proof*⟩

lemma *flow0-inj-on*:
assumes $t \leq t'$
assumes $\{t..t'\} \subseteq$ *existence-ivl0 x*
assumes $\bigwedge s.\ t < s \Longrightarrow s \leq t' \Longrightarrow$ *flow0* x $s \neq$ *flow0* x t
shows *inj-on* (*flow0* x) $\{t..t'\}$
⟨*proof*⟩

lemma *finite-ω-limit-set-in-compact-imp-unique-fixed-point*:
assumes *compact* K $K \subseteq X$
assumes $x \in X$ *trapped-forward* x K
assumes *finite* (*ω-limit-set x*)
obtains y **where** *ω-limit-set* $x = \{y\}$ $f\ y = 0$

⟨*proof*⟩

lemma *closed-orbit-periodic*:
assumes *closed-orbit x f x* $\neq$ *0*
shows *periodic-orbit x*
⟨*proof*⟩

lemma *periodic-orbitI*:
assumes $t \neq t'$ $t \in$ *existence-ivl0 y* $t' \in$ *existence-ivl0 y*
assumes *flow0 y t* = *flow0 y* t'
assumes *f y* $\neq$ *0*
shows *periodic-orbit y*
⟨*proof*⟩

lemma *periodic-orbit-recurrence-times-closed*:
assumes *periodic-orbit x*
shows *closed* $\{T \in$ *existence-ivl0 x*. $T > 0 \wedge$ *flow0 x T* = $x\}$
⟨*proof*⟩

lemma *periodic-orbit-period*:
assumes *periodic-orbit x*
shows *period x* > *0* *flow0 x* (*period x*) = *x*
⟨*proof*⟩

lemma *closed-orbit-flow0*:
assumes *closed-orbit x*
shows *closed-orbit* (*flow0 x t*)
⟨*proof*⟩

lemma *periodic-orbit-imp-flow0-regular*:
assumes *periodic-orbit x*
shows *f* (*flow0 x t*) $\neq$ *0*
⟨*proof*⟩

lemma *fixed-point-imp-*ω*-limit-set*:
assumes $x \in X$ *f x* = *0*
shows ω*-limit-set x* = $\{x\}$
⟨*proof*⟩

end

context *auto-ll-on-open* **begin**

lemma *closed-orbit-eq-rev*: *closed-orbit x* = *rev.closed-orbit x*
⟨*proof*⟩

lemma *closed-orbit-*α*-limit-set*:
assumes *closed-orbit x*
shows *flow0 x* ‘ *UNIV* = α*-limit-set x*

⟨*proof*⟩

lemma *fixed-point-imp-α-limit-set*:
assumes $x \in X$ $f\ x = 0$
shows *α-limit-set* $x = \{x\}$
⟨*proof*⟩

lemma *finite-α-limit-set-in-compact-imp-unique-fixed-point*:
assumes *compact* K $K \subseteq X$
assumes $x \in X$ *trapped-backward* x K
assumes *finite* (*α-limit-set* x)
obtains y **where** *α-limit-set* $x = \{y\}$ $f\ y = 0$
⟨*proof*⟩
end

end

6 Poincare Bendixson Theory

theory *Poincare-Bendixson*
imports
Ordinary-Differential-Equations.ODE-Analysis
Analysis-Misc ODE-Misc Periodic-Orbit
begin

6.1 Flow to Path

context *auto-ll-on-open* **begin**

definition *flow-to-path* $x\ t\ t' =$ *flow0* $x \circ$ *linepath* $t\ t'$

lemma *pathstart-flow-to-path*[*simp*]:
shows *pathstart* (*flow-to-path* $x\ t\ t'$) = *flow0* $x\ t$
⟨*proof*⟩

lemma *pathfinish-flow-to-path*[*simp*]:
shows *pathfinish* (*flow-to-path* $x\ t\ t'$) = *flow0* $x\ t'$
⟨*proof*⟩

lemma *flow-to-path-unfold*:
shows *flow-to-path* $x\ t\ t'\ s =$ *flow0* $x\ ((1 - s) * t + s * t')$
⟨*proof*⟩

lemma *subpath0-flow-to-path*:
shows (*subpath* $0\ u$ (*flow-to-path* $x\ t\ t'$)) = *flow-to-path* $x\ t\ (t + u*(t'-t))$
⟨*proof*⟩

lemma *path-image-flow-to-path*[*simp*]:

assumes $t \le t'$
shows *path-image* (*flow-to-path x t t'*) = *flow0 x* ‘ $\{t..t'\}$
⟨*proof*⟩

lemma *flow-to-path-image0-right-open*[*simp*]:
assumes $t < t'$
shows *flow-to-path x t t'* ‘ $\{0..<1\}$ = *flow0 x* ‘$\{t..<t'\}$
⟨*proof*⟩

lemma *flow-to-path-path*:
assumes $t \le t'$
assumes $\{t..t'\} \subseteq$ *existence-ivl0 x*
shows *path* (*flow-to-path x t t'*)
⟨*proof*⟩

lemma *flow-to-path-arc*:
assumes $t \le t'$
assumes $\{t..t'\} \subseteq$ *existence-ivl0 x*
assumes $\forall s \in \{t<..<t'\}$. *flow0 x s* $\neq$ *flow0 x t*
assumes *flow0 x t* $\neq$ *flow0 x t'*
shows *arc* (*flow-to-path x t t'*)
⟨*proof*⟩

end

locale *c1-on-open-R2* = *c1-on-open-euclidean f f' X* **for** *f*::*'a*::*executable-euclidean-space* ⇒ - **and** *f'* **and** *X* +
assumes *dim2*: *DIM*(*'a*) = *2*
begin

6.2 2D Line segments

Line segments are specified by two endpoints The closed line segment from x to y is given by the set x–y and x<–<y for the open segment

Rotates a vector clockwise 90 degrees

definition *rot* (*v*::*'a*) = (*eucl-of-list* [*nth-eucl v 1*, −*nth-eucl v 0*]::*'a*)

lemma *exhaust2-nat*: ($\forall i<(2::nat).\ P\ i$) $\longleftrightarrow$ $P\ 0 \land P\ 1$
⟨*proof*⟩
lemma *sum2-nat*: ($\sum i<(2::nat).\ P\ i$) = $P\ 0 + P\ 1$
⟨*proof*⟩

lemmas *vec-simps* =
eucl-eq-iff[**where** *'a=’a*] *dim2 eucl-of-list-eucl-nth exhaust2-nat*
plus-nth-eucl
minus-nth-eucl
uminus-nth-eucl
scaleR-nth-eucl

inner-nth-eucl
sum2-nat
algebra-simps

lemma *minus-expand*:
shows $(x::'a)-y = (\mathit{eucl\text{-}of\text{-}list}\ [x\$_e 0 - y\$_e 0,\ x\$_e 1 - y\$_e 1])$
⟨*proof*⟩

lemma *dot-ortho*[*simp*]: $x \cdot \mathit{rot}\ x = 0$
⟨*proof*⟩

lemma *nrm-dot*:
shows $((x::'a)-y) \cdot (\mathit{rot}\ (x-y)) = 0$
⟨*proof*⟩

lemma *nrm-reverse*: $a \cdot (\mathit{rot}\ (x-y)) = - a \cdot (\mathit{rot}\ (y-x))$ **for** $x\ y::'a$
⟨*proof*⟩

lemma *norm-rot*: $\mathit{norm}\ (\mathit{rot}\ v) = \mathit{norm}\ v$ **for** $v::'a$
⟨*proof*⟩

lemma *rot-rot*[*simp*]:
shows $\mathit{rot}\ (\mathit{rot}\ v) = -v$
⟨*proof*⟩

lemma *rot-scaleR*[*simp*]:
shows $\mathit{rot}\ (\ u *_R v) = \ u *_R (\mathit{rot}\ v)$
⟨*proof*⟩

lemma *rot-0*[*simp*]: $\mathit{rot}\ 0 = 0$
⟨*proof*⟩

lemma *rot-eq-0-iff*[*simp*]: $\mathit{rot}\ x = 0 \longleftrightarrow x = 0$
⟨*proof*⟩

lemma *in-segment-inner-rot*:
$(x - a) \cdot \mathit{rot}\ (b - a) = 0$
if $x \in \{a\text{--}b\}$
⟨*proof*⟩

lemma *inner-rot-in-segment*:
$x \in \mathit{range}\ (\lambda u.\ a + u *_R (b - a))$
if $(x - a) \cdot \mathit{rot}\ (b - a) = 0\ a \neq b$
⟨*proof*⟩

lemma *in-open-segment-iff-rot*:
$x \in \{a<\text{--}<b\} \longleftrightarrow (x - a) \cdot \mathit{rot}\ (b - a) = 0 \wedge x \cdot (b - a) \in \{a \cdot (b - a) <..< b \cdot (b - a)\}$
if $a \neq b$

⟨*proof*⟩

lemma *in-open-segment-rotD*:
$x \in \{a<--<b\} \Longrightarrow (x - a) \cdot rot\ (b - a) = 0 \wedge x \cdot (b - a) \in \{a\cdot(b - a) <..< b \cdot (b - a)\}$
⟨*proof*⟩

lemma *in-closed-segment-iff-rot*:
$x \in \{a--b\} \longleftrightarrow (x - a) \cdot rot\ (b - a) = 0 \wedge x \cdot (b - a) \in \{a\cdot(b - a) .. b \cdot (b - a)\}$
if $a \neq b$
⟨*proof*⟩

lemma *in-segment-inner-rot2*:
$(x - y) \cdot rot\ (a - b) = 0$
if $x \in \{a--b\}$ $y \in \{a--b\}$
⟨*proof*⟩

lemma *closed-segment-surface*:
$a \neq b \Longrightarrow \{a--b\} = \{ x \in \{x.\ x \cdot (b - a) \in \{a\cdot(b - a) .. b \cdot (b - a)\}\}.\ (x - a) \cdot rot\ (b - a) = 0\}$
⟨*proof*⟩

lemma *rot-diff-commute*: $rot\ (b - a) = -rot(a - b)$
⟨*proof*⟩

6.3 Bijection Real-Complex for Jordan Curve Theorem

definition *complex-of* $(x::'a) = x\$_e 0 + \mathrm{i} * x\$_e 1$

definition *real-of* $(x::complex) = (eucl\text{-}of\text{-}list\ [Re\ x,\ Im\ x]::'a)$

lemma *complex-of-linear*:
shows *linear complex-of*
⟨*proof*⟩

lemma *complex-of-bounded-linear*:
shows *bounded-linear complex-of*
⟨*proof*⟩

lemma *real-of-linear*:
shows *linear real-of*
⟨*proof*⟩

lemma *real-of-bounded-linear*:
shows *bounded-linear real-of*
⟨*proof*⟩

lemma *complex-of-real-of*:

$(complex\text{-}of \circ real\text{-}of) = id$
⟨*proof*⟩

lemma *real-of-complex-of*:
$(real\text{-}of \circ complex\text{-}of) = id$
⟨*proof*⟩

lemma *complex-of-bij*:
shows *bij* (*complex-of*)
⟨*proof*⟩

lemma *real-of-bij*:
shows *bij* (*real-of*)
⟨*proof*⟩

lemma *real-of-inj*:
shows *inj* (*real-of*)
⟨*proof*⟩

lemma *Jordan-curve-R2*:
fixes $c :: real \Rightarrow 'a$
assumes *simple-path c pathfinish c = pathstart c*
obtains *inside outside* **where**
$inside \neq \{\}$ *open inside connected inside*
$outside \neq \{\}$ *open outside connected outside*
bounded inside ¬ *bounded outside*
$inside \cap outside = \{\}$
$inside \cup outside = -\ path\text{-}image\ c$
$frontier\ inside = path\text{-}image\ c$
$frontier\ outside = path\text{-}image\ c$
⟨*proof*⟩

corollary *Jordan-inside-outside-R2*:
fixes $c :: real \Rightarrow 'a$
assumes *simple-path c pathfinish c = pathstart c*
shows $inside(path\text{-}image\ c) \neq \{\} \wedge$
$open(inside(path\text{-}image\ c)) \wedge$
$connected(inside(path\text{-}image\ c)) \wedge$
$outside(path\text{-}image\ c) \neq \{\} \wedge$
$open(outside(path\text{-}image\ c)) \wedge$
$connected(outside(path\text{-}image\ c)) \wedge$
$bounded(inside(path\text{-}image\ c)) \wedge$
$\neg\ bounded(outside(path\text{-}image\ c)) \wedge$
$inside(path\text{-}image\ c) \cap outside(path\text{-}image\ c) = \{\} \wedge$
$inside(path\text{-}image\ c) \cup outside(path\text{-}image\ c) =$
$-\ path\text{-}image\ c \wedge$
$frontier(inside(path\text{-}image\ c)) = path\text{-}image\ c \wedge$
$frontier(outside(path\text{-}image\ c)) = path\text{-}image\ c$

$\langle proof \rangle$

lemma *jordan-points-inside-outside*:
  fixes $p :: real \Rightarrow 'a$
  assumes $0 < e$
  assumes *jordan*: *simple-path p pathfinish p = pathstart p*
  assumes x: $x \in$ *path-image p*
  obtains $y\ z$ **where** $y \in$ *inside* (*path-image p*) $y \in$ *ball x e*
    $z \in$ *outside* (*path-image p*) $z \in$ *ball x e*
$\langle proof \rangle$

lemma *eventually-at-open-segment*:
  assumes $x \in \{a{<}{-}{-}{<}b\}$
  shows $\forall_F$ *y in at x*. $(y-a) \cdot rot(a-b) = 0 \longrightarrow y \in \{a <{-}{-}< b\}$
$\langle proof \rangle$

lemma *linepath-ball*:
  assumes $x \in \{a{<}{-}{-}{<}b\}$
  obtains e **where** $e > 0$ *ball x e* $\cap \{y.\ (y-a) \cdot rot(a-b) = 0\} \subseteq \{a <{-}{-}< b\}$
$\langle proof \rangle$

lemma *linepath-ball-inside-outside*:
  fixes $p :: real \Rightarrow 'a$
  assumes *jordan*: *simple-path* (p +++ *linepath a b*) *pathfinish* $p = a$ *pathstart* p $= b$
  assumes x: $x \in \{a{<}{-}{-}{<}b\}$
  obtains e **where** $e > 0$ *ball x e* $\cap$ *path-image* $p = \{\}$
    ball x e $\cap \{y.\ (y-a) \cdot rot\ (a-b) > 0\} \subseteq$ *inside* (*path-image* (p +++ *linepath* *a b*)) $\wedge$
    ball x e $\cap \{y.\ (y-a) \cdot rot\ (a-b) < 0\} \subseteq$ *outside* (*path-image* (p +++ *linepath* *a b*))
    $\vee$
    ball x e $\cap \{y.\ (y-a) \cdot rot\ (a-b) < 0\} \subseteq$ *inside* (*path-image* (p +++ *linepath* *a b*)) $\wedge$
    ball x e $\cap \{y.\ (y-a) \cdot rot\ (a-b) > 0\} \subseteq$ *outside* (*path-image* (p +++ *linepath* *a b*))
$\langle proof \rangle$

6.4 Transversal Segments

definition *transversal-segment a b* $\longleftrightarrow$
  $a \neq b \wedge \{a{-}{-}b\} \subseteq X \wedge$
  $(\forall z \in \{a{-}{-}b\}.\ f\ z \cdot rot\ (a-b) \neq 0)$

lemma *transversal-segment-reverse*:
  assumes *transversal-segment x y*
  shows *transversal-segment y x*
  $\langle proof \rangle$

lemma *transversal-segment-commute*: *transversal-segment x y* $\longleftrightarrow$ *transversal-segment y x*
⟨*proof*⟩

lemma *transversal-segment-neg*:
assumes *transversal-segment x y*
assumes *w*: $w \in \{x -- y\}$ **and** $f\ w \cdot rot\ (x-y) < 0$
shows $\forall z \in \{x--y\}.\ f(z) \cdot rot\ (x-y) < 0$
⟨*proof*⟩

lemmas *transversal-segment-sign-less* = *transversal-segment-neg*[*OF - ends-in-segment*(*1*)]

lemma *transversal-segment-pos*:
assumes *transversal-segment x y*
assumes *w*: $w \in \{x -- y\}$ $f\ w \cdot rot\ (x-y) > 0$
shows $\forall z \in \{x--y\}.\ f(z) \cdot rot\ (x-y) > 0$
⟨*proof*⟩

lemma *transversal-segment-posD*:
assumes *transversal-segment x y*
and *pos*: $z \in \{x -- y\}$ $f\ z \cdot rot\ (x - y) > 0$
shows $x \neq y$ $\{x--y\} \subseteq X$ $\bigwedge z.\ z \in \{x--y\} \Longrightarrow f\ z \cdot rot\ (x-y) > 0$
⟨*proof*⟩

lemma *transversal-segment-negD*:
assumes *transversal-segment x y*
and *pos*: $z \in \{x -- y\}$ $f\ z \cdot rot\ (x - y) < 0$
shows $x \neq y$ $\{x--y\} \subseteq X$ $\bigwedge z.\ z \in \{x--y\} \Longrightarrow f\ z \cdot rot\ (x-y) < 0$
⟨*proof*⟩

lemma *transversal-segmentE*:
assumes *transversal-segment x y*
obtains $x \neq y$ $\{x -- y\} \subseteq X$ $\bigwedge z.\ z \in \{x--y\} \Longrightarrow f\ z \cdot rot\ (x - y) > 0$
| $x \neq y$ $\{x -- y\} \subseteq X$ $\bigwedge z.\ z \in \{x--y\} \Longrightarrow f\ z \cdot rot\ (y - x) > 0$
⟨*proof*⟩

lemma *dist-add-vec*:
shows $dist\ (x + s *_R v)\ x = abs\ s * norm\ v$
⟨*proof*⟩

lemma *transversal-segment-exists*:
assumes $x \in X$ $f\ x \neq 0$
obtains *a b* **where** $x \in \{a<--<b\}$
transversal-segment a b
⟨*proof*⟩

Perko Section 3.7 Lemma 2 part 1.

lemma *flow-transversal-segment-finite-intersections*:
assumes *transversal-segment a b*

assumes $t \le t'$ $\{t\ ..\ t'\} \subseteq$ *existence-ivl0 x*
shows *finite* $\{s{\in}\{t..t'\}.\ \mathit{flow0}\ x\ s \in \{a{-}{-}b\}\}$
⟨*proof*⟩

lemma *transversal-bound-posE*:
assumes *transversal*: *transversal-segment a b*
assumes *direction*: $z \in \{a\ {-}{-}\ b\}$ $f\ z \cdot (\mathit{rot}\ (a - b)) > 0$
obtains $d\ B$ **where** $d > 0$ $0 < B$
$\bigwedge x\ y.\ x \in \{a\ {-}{-}\ b\} \Longrightarrow \mathit{dist}\ x\ y \le d \Longrightarrow f\ y \cdot (\mathit{rot}\ (a - b)) \ge B$
⟨*proof*⟩

lemma *transversal-bound-negE*:
assumes *transversal*: *transversal-segment a b*
assumes *direction*: $z \in \{a\ {-}{-}\ b\}$ $f\ z \cdot (\mathit{rot}\ (a - b)) < 0$
obtains $d\ B$ **where** $d > 0$ $0 < B$
$\bigwedge x\ y.\ x \in \{a\ {-}{-}\ b\} \Longrightarrow \mathit{dist}\ x\ y \le d \Longrightarrow f\ y \cdot (\mathit{rot}\ (b - a)) \ge B$
⟨*proof*⟩

lemma *leaves-transversal-segmentE*:
assumes *transversal*: *transversal-segment a b*
obtains $T\ n$ **where** $T > 0$ $n = a - b \lor n = b - a$
$\bigwedge x.\ x \in \{a\ {-}{-}\ b\} \Longrightarrow \{-T..T\} \subseteq$ *existence-ivl0 x*
$\bigwedge x\ s.\ x \in \{a\ {-}{-}\ b\} \Longrightarrow 0 < s \Longrightarrow s \le T \Longrightarrow$
$(\mathit{flow0}\ x\ s - x) \cdot \mathit{rot}\ n > 0$
$\bigwedge x\ s.\ x \in \{a\ {-}{-}\ b\} \Longrightarrow -T \le s \Longrightarrow s < 0 \Longrightarrow$
$(\mathit{flow0}\ x\ s - x) \cdot \mathit{rot}\ n < 0$
⟨*proof*⟩

lemma *inner-rot-pos-move-base*: $(x - a) \cdot \mathit{rot}\ (a - b) > 0$
if $(x - y) \cdot \mathit{rot}\ (a - b) > 0$ $y \in \{a\ {-}{-}\ b\}$
⟨*proof*⟩

lemma *inner-rot-neg-move-base*: $(x - a) \cdot \mathit{rot}\ (a - b) < 0$
if $(x - y) \cdot \mathit{rot}\ (a - b) < 0$ $y \in \{a\ {-}{-}\ b\}$
⟨*proof*⟩

lemma *inner-pos-move-base*: $(x - a) \cdot n > 0$
if $(a - b) \cdot n = 0$ $(x - y) \cdot n > 0$ $y \in \{a\ {-}{-}\ b\}$
⟨*proof*⟩

lemma *inner-neg-move-base*: $(x - a) \cdot n < 0$
if $(a - b) \cdot n = 0$ $(x - y) \cdot n < 0$ $y \in \{a\ {-}{-}\ b\}$
⟨*proof*⟩

lemma *rot-same-dir*:
assumes $x1 \in \{a{<}{-}{-}{<}b\}$
assumes $x2 \in \{x1{<}{-}{-}{<}b\}$
shows $(y \cdot \mathit{rot}\ (a{-}b) > 0) = (y \cdot \mathit{rot}(x1{-}x2) > 0)$ $(y \cdot \mathit{rot}\ (a{-}b) < 0) = (y \cdot$

$rot(x1-x2) < 0)$
⟨proof⟩

6.5 Monotone Step Lemma

lemma *flow0-transversal-segment-monotone-step*:
assumes *transversal-segment a b*
assumes $t1 \le t2$ $\{t1..t2\} \subseteq$ *existence-ivl0 x*
assumes *x1*: *flow0 x t1* $\in \{a<\!\!-\!\!-\!\!<b\}$
assumes *x2*: *flow0 x t2* $\in \{$*flow0 x t1*$<\!\!-\!\!-\!\!<b\}$
assumes $\bigwedge t.\ t \in \{t1<..<t2\} \Longrightarrow$ *flow0 x t* $\notin \{a<\!\!-\!\!-\!\!<b\}$
assumes $t > t2$ $t \in$ *existence-ivl0 x*
shows *flow0 x t* $\notin \{a<\!\!-\!\!-\!\!<$*flow0 x t2*$\}$
⟨proof⟩

lemma *open-segment-trichotomy*:
fixes $x\ y\ a\ b::'a$
assumes x:$x \in \{a<\!\!-\!\!-\!\!<b\}$
assumes y:$y \in \{a<\!\!-\!\!-\!\!<b\}$
shows $x = y \vee y \in \{x<\!\!-\!\!-\!\!<b\} \vee y \in \{a<\!\!-\!\!-\!\!<x\}$
⟨proof⟩

sublocale *rev*: *c1-on-open-R2* $-f$ $-f'$ **rewrites** $-(-f) = f$ **and** $-(-f') = f'$
⟨proof⟩

lemma *rev-transversal-segment*: *rev.transversal-segment a b* = *transversal-segment a b*
⟨proof⟩

lemma *flow0-transversal-segment-monotone-step-reverse*:
assumes *transversal-segment a b*
assumes $t1 \le t2$
assumes $\{t1..t2\} \subseteq$ *existence-ivl0 x*
assumes *x1*: *flow0 x t1* $\in \{a<\!\!-\!\!-\!\!<b\}$
assumes *x2*: *flow0 x t2* $\in \{a<\!\!-\!\!-\!\!<$*flow0 x t1*$\}$
assumes $\bigwedge t.\ t \in \{t1<..<t2\} \Longrightarrow$ *flow0 x t* $\notin \{a<\!\!-\!\!-\!\!<b\}$
assumes $t < t1$ $t \in$ *existence-ivl0 x*
shows *flow0 x t* $\notin \{a<\!\!-\!\!-\!\!<$*flow0 x t1*$\}$
⟨proof⟩

lemma *flow0-transversal-segment-monotone-step-reverse2*:
assumes *transversal*: *transversal-segment a b*
assumes *time*: $t1 \le t2$
assumes *exist*: $\{t1..t2\} \subseteq$ *existence-ivl0 x*
assumes *t1*: *flow0 x t1* $\in \{a<\!\!-\!\!-\!\!<b\}$
assumes *t2*: *flow0 x t2* $\in \{$*flow0 x t1*$<\!\!-\!\!-\!\!<b\}$
assumes *t1t2*: $\bigwedge t.\ t \in \{t1<..<t2\} \Longrightarrow$ *flow0 x t* $\notin \{a<\!\!-\!\!-\!\!<b\}$
assumes *t*: $t < t1$ $t \in$ *existence-ivl0 x*
shows *flow0 x t* $\notin \{$*flow0 x t1*$<\!\!-\!\!-\!\!<b\}$

$\langle proof \rangle$

lemma *flow0-transversal-segment-monotone-step2*:
  assumes *transversal*: *transversal-segment a b*
  assumes *time*: $t1 \leq t2$
  assumes *exist*: $\{t1..t2\} \subseteq$ *existence-ivl0 x*
  assumes *t1*: *flow0 x t1* $\in \{a{<}{-}{-}{<}b\}$
  assumes *t2*: *flow0 x t2* $\in \{a{<}{-}{-}{<}$*flow0 x t1*$\}$
  assumes *t1t2*: $\bigwedge t.\ t \in \{t1{<}..{<}t2\} \Longrightarrow$ *flow0 x t* $\notin \{a{<}{-}{-}{<}b\}$
  shows $\bigwedge t.\ t > t2 \Longrightarrow t \in$ *existence-ivl0 x* $\Longrightarrow$ *flow0 x t* $\notin \{$*flow0 x t2*${<}{-}{-}{<}b\}$
  $\langle proof \rangle$

lemma *flow0-transversal-segment-monotone*:
  assumes *transversal-segment a b*
  assumes $t1 \leq t2$
  assumes $\{t1..t2\} \subseteq$ *existence-ivl0 x*
  assumes *x1*: *flow0 x t1* $\in \{a{<}{-}{-}{<}b\}$
  assumes *x2*: *flow0 x t2* $\in \{$*flow0 x t1*${<}{-}{-}{<}b\}$
  assumes $t > t2$ $t \in$ *existence-ivl0 x*
  shows *flow0 x t* $\notin \{a{<}{-}{-}{<}$*flow0 x t2*$\}$
$\langle proof \rangle$

6.6 Straightening

This lemma uses the implicit function theorem

lemma *cross-time-continuous*:
  assumes *transversal-segment a b*
  assumes $x \in \{a{<}{-}{-}{<}b\}$
  assumes $e > 0$
  obtains *d t* **where** $d > 0$ *continuous-on* (*ball x d*) *t*
    $\bigwedge y.\ y \in$ *ball x d* $\Longrightarrow$ *flow0 y* $(t\ y) \in \{a{<}{-}{-}{<}b\}$
    $\bigwedge y.\ y \in$ *ball x d* $\Longrightarrow |t\ y| < e$
    continuous-on (*ball x d*) *t*
    $t\ x = 0$
$\langle proof \rangle$

lemma *ω-limit-crossings*:
  assumes *transversal-segment a b*
  assumes *pos-ex*: $\{0..\} \subseteq$ *existence-ivl0 x*
  assumes *ω-limit-point x p*
  assumes $p \in \{a{<}{-}{-}{<}b\}$
  obtains *s* **where**
    $s \longrightarrow \infty$
    (*flow0 x* $\circ$ *s*) $\longrightarrow p$
    $\forall_F$ *n in sequentially. flow0 x* $(s\ n) \in \{a{<}{-}{-}{<}b\} \wedge s\ n \in$ *existence-ivl0 x*
$\langle proof \rangle$

lemma *filterlim-at-top-tendstoE*:

assumes $e > 0$
assumes *filterlim s at-top sequentially*
assumes $(flow0\ x \circ s) \longrightarrow u$
assumes $\forall_F\ n$ *in sequentially*. $P\ (s\ n)$
obtains m **where** $m > b$ $P\ m$ $dist\ (flow0\ x\ m)\ u < e$
⟨*proof*⟩

lemma *open-segment-separate-left*:
fixes $u\ v\ x\ a\ b{::}'a$
assumes $u{:}u \in \{a <{-}{-}< b\}$
assumes $v{:}v \in \{u <{-}{-}< b\}$
assumes x: $dist\ x\ u < dist\ u\ v$ $x \in \{a <{-}{-}< b\}$
shows $x \in \{a <{-}{-}< v\}$
⟨*proof*⟩

lemma *open-segment-separate-right*:
fixes $u\ v\ x\ a\ b{::}'a$
assumes $u{:}u \in \{a <{-}{-}< b\}$
assumes $v{:}v \in \{a <{-}{-}< u\}$
assumes x: $dist\ x\ u < dist\ u\ v$ $x \in \{a <{-}{-}< b\}$
shows $x \in \{v <{-}{-}< b\}$
⟨*proof*⟩

lemma *no-two-ω-limit-points*:
assumes *transversal*: *transversal-segment a b*
assumes *ex-pos*: $\{0..\} \subseteq$ *existence-ivl0 x*
assumes u: *ω-limit-point x u* $u \in \{a<{-}{-}<b\}$
assumes v: *ω-limit-point x v* $v \in \{a<{-}{-}<b\}$
assumes *uv*: $v \in \{u<{-}{-}<b\}$
shows *False*
⟨*proof*⟩

6.7 Unique Intersection

Perko Section 3.7 Remark 2

lemma *unique-transversal-segment-intersection*:
assumes *transversal-segment a b*
assumes $\{0..\} \subseteq$ *existence-ivl0 x*
assumes $u \in$ *ω-limit-set x* $\cap \{a<{-}{-}<b\}$
shows *ω-limit-set x* $\cap \{a<{-}{-}<b\} = \{u\}$
⟨*proof*⟩

Adapted from Perko Section 3.7 Lemma 4 (+ Chicone)

lemma *periodic-imp-ω-limit-set*:
assumes *compact K* $K \subseteq X$
assumes $x \in X$ *trapped-forward x K*
assumes *periodic-orbit y*
flow0 y ‘ UNIV $\subseteq$ *ω-limit-set x*
shows *flow0 y ‘UNIV* $=$ *ω-limit-set x*

$\langle proof \rangle$

end context *c1-on-open-R2* **begin**

lemma *α-limit-crossings*:
  assumes *transversal-segment a b*
  assumes *pos-ex*: $\{..0\} \subseteq$ *existence-ivl0 x*
  assumes *α-limit-point x p*
  assumes $p \in \{a{<}{-}{-}{<}b\}$
  obtains s **where**
    $s \longrightarrow -\infty$
    (*flow0 x* $\circ$ s) $\longrightarrow p$
    $\forall_F$ *n in sequentially.*
    flow0 x $(s\ n) \in \{a{<}{-}{-}{<}b\} \wedge$
    $s\ n \in$ *existence-ivl0 x*
$\langle proof \rangle$

If a positive limit point has a regular point in its positive limit set then it is periodic

lemma *ω-limit-point-ω-limit-set-regular-imp-periodic*:
  assumes *compact K* $K \subseteq X$
  assumes $x \in X$ *trapped-forward x K*
  assumes *y*: $y \in$ *ω-limit-set x* $f\ y \neq 0$
  assumes *z*: $z \in$ *ω-limit-set y* $\cup$ *α-limit-set y* $f\ z \neq 0$
  shows *periodic-orbit y* $\wedge$ *flow0 y ‘ UNIV* $=$ *ω-limit-set x*
$\langle proof \rangle$

6.8 Poincare Bendixson Theorems

Perko Section 3.7 Theorem 1

theorem *poincare-bendixson*:
  assumes *compact K* $K \subseteq X$
  assumes $x \in X$ *trapped-forward x K*
  assumes $0 \notin f$ ‘ (*ω-limit-set x*)
  obtains *y* **where** *periodic-orbit y*
    flow0 y ‘ UNIV $=$ *ω-limit-set x*
$\langle proof \rangle$

lemma *fixed-point-in-ω-limit-set-imp-ω-limit-set-singleton-fixed-point*:
  assumes *compact K* $K \subseteq X$
  assumes $x \in X$ *trapped-forward x K*
  assumes *fp*: $yfp \in$ *ω-limit-set x* $f\ yfp = 0$
  assumes *zpx*: $z \in$ *ω-limit-set x*
  assumes *finite-fp*: *finite* $\{y \in K.\ f\ y = 0\}$ (**is** *finite ?S*)
  shows ($\exists\, p1 \in$ *ω-limit-set x*. $f\ p1 = 0 \wedge$ *ω-limit-set z* $= \{p1\}$) $\wedge$
    ($\exists\, p2 \in$ *ω-limit-set x*. $f\ p2 = 0 \wedge$ *α-limit-set z* $= \{p2\}$)
$\langle proof \rangle$

end context *c1-on-open-R2* **begin**

Perko Section 3.7 Theorem 2

theorem *poincare-bendixson-general*:
  assumes *compact* K $K \subseteq X$
  assumes $x \in X$ *trapped-forward* x K
  assumes $S = \{y \in K.\ f\ y = 0\}$ *finite* S
  shows
    $(\exists\, y \in S.\ \omega\text{-}limit\text{-}set\ x = \{y\}) \vee$
  $(\exists\, y.\ periodic\text{-}orbit\ y \wedge$
    $flow0\ y\ \text{‘}\ UNIV = \omega\text{-}limit\text{-}set\ x) \vee$
  $(\exists\, P\ R.\ \omega\text{-}limit\text{-}set\ x = P \cup R \wedge$
    $P \subseteq S \wedge 0 \notin f\ \text{‘}\ R \wedge R \neq \{\} \wedge$
    $(\forall\, z \in R.$
      $(\exists\, p1 \in P.\ \omega\text{-}limit\text{-}set\ z = \{p1\}) \wedge$
      $(\exists\, p2 \in P.\ \alpha\text{-}limit\text{-}set\ z = \{p2\})))$
⟨*proof*⟩

corollary *poincare-bendixson-applied*:
  assumes *compact* K $K \subseteq X$
  assumes $K \neq \{\}$ *positively-invariant* K
  assumes $0 \notin f\ \text{‘}\ K$
  obtains y **where** *periodic-orbit* y *flow0* y ‘ $UNIV \subseteq K$
⟨*proof*⟩

definition *limit-cycle* $y \longleftrightarrow$
  $periodic\text{-}orbit\ y \wedge$
  $(\exists\, x.\ x \notin flow0\ y\ \text{‘}\ UNIV \wedge$
  $(flow0\ y\ \text{‘}\ UNIV = \omega\text{-}limit\text{-}set\ x \vee flow0\ y\ \text{‘}\ UNIV = \alpha\text{-}limit\text{-}set\ x))$

corollary *poincare-bendixson-limit-cycle*:
  assumes *compact* K $K \subseteq X$
  assumes $x \in K$ *positively-invariant* K
  assumes $0 \notin f\ \text{‘}\ K$
  assumes $rev.flow0\ x\ t \notin K$
  obtains y **where** *limit-cycle* y *flow0* y ‘ $UNIV \subseteq K$
⟨*proof*⟩

end

end
theory *Affine-Arithmetic-Misc*
  imports *HOL−ODE−Numerics.ODE-Numerics*
begin

7 Branch-And-Bound Arithmetic

primrec *prove-nonneg*::(*nat* ∗ *nat* ∗ *string*) *list* ⇒ *nat* ⇒ *nat* ⇒ *slp* ⇒ *real aform list list* ⇒ *bool* **where**
prove-nonneg prnt 0 p slp X = (*let* - = *if prnt* ≠ [] *then print* (*STR* ''# *depth limit exceeded*$\boxed{\hookleftarrow}$'') *else* () *in False*)
| *prove-nonneg prnt* (*Suc i*) *p slp XXS* =
(*case XXS of* [] ⇒ *True* | (*X*#*XS*) ⇒
let RS = *approx-slp-outer p 1 slp X*
in if RS≠*None* ∧ *Inf-aform*′ *p* (*hd* (*the RS*)) ≥ *0*
then
let - = *if prnt* ≠ [] *then print* (*STR* ''# *Success*$\boxed{\hookleftarrow}$'') *else* ();
- = *if prnt* ≠ [] *then print* (*String.implode* ((*shows* ''# '' *o shows-box-of-aforms-hr X*) ''$\boxed{\hookleftarrow}$'')) *else* ();
- = *fold* (λ(*a*, *b*, *c*) -. *print* (*String.implode* (*shows-segments-of-aform a b X c* ''$\boxed{\hookleftarrow}$''))) *prnt* ()
in prove-nonneg prnt i p slp XS
else let - = *if prnt* ≠ [] *then print* (*STR* ''# *Split*$\boxed{\hookleftarrow}$'') *else* () *in case split-aforms-largest-uncond X of* (*a*, *b*) ⇒
prove-nonneg prnt i p slp (*a*#*b*#*XS*))

lemma *prove-nonneg-simps*[*simp*]:
prove-nonneg prnt 0 p slp X = *False*
prove-nonneg prnt (*Suc i*) *p slp XXS* =
(*case XXS of* [] ⇒ *True* | (*X*#*XS*) ⇒
let RS = *approx-slp-outer p 1 slp X*
in if RS≠*None* ∧ *Inf-aform*′ *p* (*hd* (*the RS*)) ≥ *0*
then prove-nonneg prnt i p slp XS
else case split-aforms-largest-uncond X of (*a*, *b*) ⇒ *prove-nonneg prnt i p slp* (*a*#*b*#*XS*))
⟨*proof*⟩

lemmas [*simp del*] = *prove-nonneg.simps*

lemma *split-aforms-lemma*:
fixes *xs*::*real list*
assumes *split-aforms XS i* = (*YS*, *ZS*)
assumes *xs* ∈ *Joints XS*
shows *xs* ∈ *Joints YS* ∪ *Joints ZS*
⟨*proof*⟩

lemma *prove-nonneg-empty*[*simp*]: *prove-nonneg prnt* (*Suc i*) *p slp* []
⟨*proof*⟩

lemma *prove-nonneg-fuel-mono*:
prove-nonneg prnt (*Suc i*) *p* (*slp-of-fas* [*fa*]) *YSS*
if *prove-nonneg prnt i p* (*slp-of-fas* [*fa*]) *YSS*
⟨*proof*⟩

lemma *prove-nonneg-mono*:
prove-nonneg prnt i p (*slp-of-fas* [*fa*]) *YSS* **if** *prove-nonneg prnt i p* (*slp-of-fas* [*fa*]) (*YS* # *YSS*)
⟨*proof*⟩

lemma *prove-nonneg*:
assumes *prove-nonneg prnt i p* (*slp-of-fas* [*fa*]) *XSS*
shows ∀ *XS* ∈ *set XSS*. ∀ *xs* ∈ *Joints XS*. *interpret-floatarith fa xs* ≥ *0*
⟨*proof*⟩

end

8 Examples

theory *Examples*
imports *Poincare-Bendixson*
HOL−ODE−Numerics.ODE-Numerics
Affine-Arithmetic-Misc
begin

8.1 Simple

context
begin

coordinate functions

definition *cx x y* = −*y* + *x* ∗ (*1* − *x*^*2* − *y*^*2*)
definition *cy x y* = *x* + *y* ∗ (*1* − *x*^*2* − *y*^*2*)

lemmas *c-defs* = *cx-def cy-def*

partial derivatives

definition *C11*::*real*⇒*real*⇒*real* **where** *C11 x y* = *1* − *3* ∗ *x*^*2* − *y*^*2*
definition *C12*::*real*⇒*real*⇒*real* **where** *C12 x y* = −*1* − *2* ∗ *x* ∗ *y*
definition *C21*::*real*⇒*real*⇒*real* **where** *C21 x y* = *1* − *2* ∗ *x* ∗ *y*
definition *C22*::*real*⇒*real*⇒*real* **where** *C22 x y* = *1* − *x*^*2* − *3* ∗ *y*^*2*

lemmas *C-partials* = *C11-def C12-def C21-def C22-def*

Jacobian as linear map

definition *C* :: *real* ⇒ *real* ⇒ (*real* × *real*) $\Rightarrow_L$ (*real* × *real*) **where**
C x y = *blinfun-of-matrix*
((λ- -. *0*)
((*1*,*0*) := (λ-. *0*)((*1*, *0*):=*C11 x y*, (*0*, *1*):=*C12 x y*),
(*0*, *1*):= (λ-. *0*)((*1*, *0*):=*C21 x y*, (*0*, *1*):=*C22 x y*)))

lemma *C-simp*[*simp*]: *blinfun-apply* (*C x y*) (*dx*, *dy*) =
(*dx* ∗ *C11 x y* + *dy* ∗ *C12 x y*,

dx ∗ *C21 x y* + *dy* ∗ *C22 x y*)
⟨*proof*⟩

lemma *C-continuous*[*continuous-intros*]:
continuous-on S (λ*x. local.C* (*f x*) (*g x*))
if *continuous-on S f continuous-on S g*
⟨*proof*⟩

interpretation *c*: *c1-on-open-R2* λ(*x*::*real*, *y*::*real*). (*cx x y*, *cy x y*)::*real*∗*real*
λ(*x*, *y*). *C x y UNIV*
⟨*proof*⟩

definition *trapC* = *cball* (*0*::*real*,*0*::*real*) *2* − *ball* (*0*::*real*,*0*::*real*) (*1*/*2*)

lemma *trapC-eq*:
shows *trapC* = {*p*. (*fst p*)^*2* + (*snd p*)^*2* − *4* ≤ *0*} ∩ {*p*. *1*/*4* − ((*fst p*)^*2* + (*snd p*)^*2*) ≤ *0*}
⟨*proof*⟩

lemma *x-in-trapC*:
shows (*2*,*0*) ∈ *trapC*
⟨*proof*⟩

lemma *compact-trapC*:
shows *compact trapC*
⟨*proof*⟩

lemma *nonempty-trapC*:
shows *trapC* ≠ {}
⟨*proof*⟩

lemma *origin-fixpoint*:
assumes (λ(*x*, *y*). (*cx x y*, *cy x y*)) (*a*,*b*) = *0*
shows *a* = (*0*::*real*) *b* = (*0*::*real*)
⟨*proof*⟩

lemma *origin-not-trapC*:
shows *0* ∉ *trapC*
⟨*proof*⟩

lemma *regular-trapC*:
shows *0* ∉ (λ(*x*, *y*). (*cx x y*, *cy x y*)) ‘ *trapC*
⟨*proof*⟩

lemma *positively-invariant-outer*:
shows *c.positively-invariant* {*p*. (λ*p*. (*fst p*)^2 + (*snd p*)^2 − *4*) *p* ≤ *0*}
⟨*proof*⟩

lemma *positively-invariant-inner*:
shows *c.positively-invariant* $\{p.\ (\lambda p.\ 1/4 - ((fst\ p)^2 + (snd\ p)^2))\ p \le 0\}$
⟨*proof*⟩

lemma *positively-invariant-trapC*:
shows *c.positively-invariant trapC*
⟨*proof*⟩

theorem *c-has-periodic-orbit*:
obtains *y* **where** *c.periodic-orbit y c.flow0 y ‘ UNIV* $\subseteq$ *trapC*
⟨*proof*⟩

Real-Arithmetic

schematic-goal *c-fas*:
$[-(-(X!1) + (X!0) * (1 - (X!0)\hat{}2 - (X!1)\hat{}2)), -((X!0) + (X!1) * (1 - (X!0)\hat{}2 - (X!1)\hat{}2))] = interpret\text{-}floatariths\ ?fas\ X$
⟨*proof*⟩

concrete-definition *c-fas* **uses** *c-fas*

interpretation *crev*: *ode-interpretation true-form UNIV c-fas*
$-(\lambda(x, y).\ (cx\ x\ y,\ cy\ x\ y)::real*real)$
$d::2$ **for** d
⟨*proof*⟩

lemma *crev*: $t \in \{1/8\ ..\ 1/8\} \longrightarrow (x, y) \in \{(2, 0)\ ..\ (2, 0)\} \longrightarrow$
$t \in c.rev.existence\text{-}ivl0\ (x, y) \wedge c.rev.flow0\ (x, y)\ t \in \{(5.15, -0.651)..(5.18, -0.647)\}$
⟨*proof*⟩

theorem *c-has-limit-cycle*:
obtains *y* **where** *c.limit-cycle y range* (*c.flow0 y*) $\subseteq$ *trapC*
⟨*proof*⟩

end

8.2 Glycolysis

Strogatz, Example 7.3.2

context
begin

coordinate functions

definition $gx\ x\ y = -x + 0.08 * y + x^2 * y$
definition $gy\ x\ y = 0.6 - 0.08 * y - x^2 * y$

lemmas *g-defs = gx-def gy-def*

partial derivatives

definition $A11$::$real \Rightarrow real \Rightarrow real$ **where** $A11\ x\ y = -1 + 2 * x * y$
definition $A12$::$real \Rightarrow real \Rightarrow real$ **where** $A12\ x\ y = (0.08 + x^2)$
definition $A21$::$real \Rightarrow real \Rightarrow real$ **where** $A21\ x\ y = -2 * x * y$
definition $A22$::$real \Rightarrow real \Rightarrow real$ **where** $A22\ x\ y = -(0.08 + x^2)$

lemmas *A-partials = A11-def A12-def A21-def A22-def*

Jacobian as linear map

definition A :: $real \Rightarrow real \Rightarrow (real \times real) \Rightarrow_L (real \times real)$ **where**
$A\ x\ y$ = *blinfun-of-matrix*
$((\lambda\text{- -}.\ 0)$
$((1,0) := (\lambda\text{-}.\ 0)((1,\ 0):=A11\ x\ y,\ (0,\ 1):=A12\ x\ y),$
$(0,\ 1):= (\lambda\text{-}.\ 0)((1,\ 0):=A21\ x\ y,\ (0,\ 1):=A22\ x\ y)))$

lemma *A-simp*[*simp*]: *blinfun-apply* $(A\ x\ y)\ (dx,\ dy) =$
$(dx * A11\ x\ y + dy * A12\ x\ y,$
$dx * A21\ x\ y + dy * A22\ x\ y)$
⟨*proof*⟩

lemma *A-continuous*[*continuous-intros*]:
continuous-on S (λx. *local.A* $(f\ x)\ (g\ x)$)
if *continuous-on* S f *continuous-on* S g
⟨*proof*⟩

interpretation g: *c1-on-open-R2* $\lambda(x$::*real*, y::*real*). $(gx\ x\ y,\ gy\ x\ y)$::*real*∗*real*
$\lambda(x,\ y).\ A\ x\ y$ *UNIV*
⟨*proof*⟩

definition (*pos-quad*::(*real* × *real*) *set*) = $\{p\ .\ -\ snd\ p \leq 0\} \cap \{p\ .\ -\ fst\ p \leq 0\}$

definition (*trapG1*::(*real* × *real*) *set*) = *pos-quad* ∩ ($\{p.\ (snd\ p) - 751/100 \leq 0\}$
$\cap\ \{p.\ (fst\ p) + (snd\ p) - 812/100 \leq 0\}$)

lemma *positively-invariant-y*:
shows *g.positively-invariant* $\{p\ .\ -\ snd\ p \leq 0\}$
⟨*proof*⟩

lemma *positively-invariant-pos-quad*:
shows *g.positively-invariant pos-quad*
⟨*proof*⟩

lemma *positively-invariant-y-upper*:
shows *g.positively-invariant* $\{p.\ (snd\ p) - 751/100 \leq 0\}$
⟨*proof*⟩

lemma *arith2*:
shows $(y$::*real*$) \leq 751/100 \wedge x + (y$::*real*$) = 812/100 \Longrightarrow 3/5 - (x$::*real*$) < 0$
⟨*proof*⟩

lemma *positively-invariant-trapG1*:
  shows *g.positively-invariant trapG1*
  ⟨*proof*⟩

definition *p1* (*x*::*real*) (*y*::*real*) = $-(21/34) - (69*x)/38 + (19*x\hat{~}2)/15 - (9*x\hat{~}3)/28 - (6*x\hat{~}4)/43 + (\ 14*y)/29 + (31*x*y)/21 + (182*x\hat{~}2*y)/47 - (35*x\hat{~}3*y)/16 - (\ 3*y\hat{~}2)/17 - (2*x*y\hat{~}2)/9 - (31*x\hat{~}2*y\hat{~}2)/20 + y\hat{~}3/102 + (x*y\hat{~}3)/59$

definition *p1d x xa* = $38 * (\mathit{fst}\ xa * \mathit{fst}\ x) \ / \ 15 - 69 * \mathit{fst}\ xa \ / \ 38 -$
$27 * (\mathit{fst}\ xa * (\mathit{fst}\ x)^2) \ / \ 28 -$
$24 * (\mathit{fst}\ xa * \mathit{fst}\ x \ \hat{}\ 3) \ / \ 43 +$
$14 * \mathit{snd}\ xa \ / \ 29 +$
$(651 * (\mathit{fst}\ x * \mathit{snd}\ xa) +$
$651 * (\mathit{fst}\ xa * \mathit{snd}\ x)) \ /$
$441 +$
$(8554 * ((\mathit{fst}\ x)^2 * \mathit{snd}\ xa) +$
$17108 * (\mathit{fst}\ xa * (\mathit{fst}\ x * \mathit{snd}\ x))) \ /$
$2209 -$
$(560 * (\mathit{fst}\ x \ \hat{}\ 3 * \mathit{snd}\ xa) +$
$1680 * (\mathit{fst}\ xa * ((\mathit{fst}\ x)^2 * \mathit{snd}\ x))) \ /$
$256 -$
$6 * (\mathit{snd}\ xa * \mathit{snd}\ x) \ / \ 17 -$
$(36 * (\mathit{fst}\ x * (\mathit{snd}\ xa * \mathit{snd}\ x)) +$
$18 * (\mathit{fst}\ xa * (\mathit{snd}\ x)^2)) \ /$
$81 -$
$(1240 * ((\mathit{fst}\ x)^2 * (\mathit{snd}\ xa * \mathit{snd}\ x)) +$
$1240 * (\mathit{fst}\ xa * (\mathit{fst}\ x * (\mathit{snd}\ x)^2))) \ /$
$400 +$
$\mathit{snd}\ xa * (\mathit{snd}\ x)^2 \ / \ 34 +$
$(177 * (\mathit{fst}\ x * (\mathit{snd}\ xa * (\mathit{snd}\ x)^2)) +$
$\mathit{fst}\ xa * \mathit{snd}\ x \ \hat{}\ 3 * 59) \ /$
3481

lemma *p1-has-derivative*:
  shows ((λx. *p1* (*fst x*) (*snd x*)) *has-derivative p1d x*) (*at x*)
  ⟨*proof*⟩

lemma *p1-not-equil*:
  shows $p1\ x\ y \leq 0 \Longrightarrow gx\ x\ y \neq 0 \lor gy\ x\ y \neq 0$
  ⟨*proof*⟩

definition *trapG* = $\mathit{trapG1} \cap \{p.\ p1\ (\mathit{fst}\ p)\ (\mathit{snd}\ p) \leq 0\}$

Real-Arithmetic

definition *g-arith a b* = $(-\ (27\ /\ 25) - a^2 + 2 * a * b) * p1\ a\ b - p1d\ (a,\ b)$

$(gx\ a\ b,\ gy\ a\ b)$

schematic-goal *g-arith-fas*:
$[g\text{-}arith\ (X!0)\ (X!1)] = interpret\text{-}floatariths\ ?fas\ X$
⟨*proof*⟩

concrete-definition *g-arith-fas* **uses** *g-arith-fas*

lemma *list-interval2*: $list\text{-}interval\ [a,\ b]\ [c,\ d] = \{[x,\ y] \mid x\ y.\ x \in \{a\ ..\ c\} \wedge y \in \{b\ ..\ d\}\}$
⟨*proof*⟩

lemma *g-arith-nonneg*: $g\text{-}arith\ a\ b \geq 0$
if a: $0 \leq a\ a \leq 8.24$ **and** b: $0 \leq b\ b \leq 7.51$
⟨*proof*⟩

lemma *trap-arithmetic*:
$p1d\ (a,\ b)\ (gx\ a\ b,\ gy\ a\ b) \leq (-\ (27\ /\ 25) - a^2 + 2 * a * b) * p1\ a\ b$ **if** $(a,\ b) \in trapG1$
⟨*proof*⟩

lemma *positively-invariant-trapG*:
shows *g.positively-invariant trapG*
⟨*proof*⟩

lemma *regular-trapG*:
shows $0 \notin (\lambda(x,\ y).\ (gx\ x\ y,\ gy\ x\ y))\ `\ trapG$
⟨*proof*⟩

lemma *arith*:
$\bigwedge a\ b{::}real.\ 0 \leq b \Longrightarrow$
$0 \leq a \Longrightarrow$
$b * 100 \leq 751 \Longrightarrow$
$a * 25 + b * 25 \leq 203 \Longrightarrow norm\ a + norm\ b \leq 20$
⟨*proof*⟩

lemma *trapG1-subset*:
shows $trapG1 \subseteq cball\ (0{::}real \times real)\ 20$
⟨*proof*⟩

lemma *compact-subset-closed*:
assumes *compact S closed T*
assumes $T \subseteq S$
shows *compact T*
⟨*proof*⟩

lemma *compact-trapG1*:
shows *compact trapG1*
⟨*proof*⟩

lemma *compact-trapG*:
  shows *compact trapG*
  ⟨*proof*⟩

lemma *x-in-trapG*:
  shows $(1,0) \in trapG$
  ⟨*proof*⟩

schematic-goal *g-fas*:
  $[- (- (X!0) + 8 \ / \ 100 * (X!1) + (X!0)\hat{}2 * (X!1)),-(6 \ / \ 10 - 8 \ / \ 100 * (X!1) - (X!0)\hat{}2 * (X!1))] = interpret\text{-}floatariths\ ?fas\ X$
  ⟨*proof*⟩

concrete-definition *g-fas* **uses** *g-fas*

interpretation *grev*: *ode-interpretation true-form UNIV g-fas*
  $-(\lambda(x, y). (gx\ x\ y,\ gy\ x\ y)::real*real)$
  $d::2$ **for** d
  ⟨*proof*⟩

lemma *grev*: $t \in \{1/8 .. 1/8\} \longrightarrow (x, y) \in \{(1, 0) .. (1, 0)\} \longrightarrow$
  $t \in g.rev.existence\text{-}ivl0\ (x, y) \wedge g.rev.flow0\ (x, y)\ t \in$
  $\{(1.1, -0.09) .. (1.2, -0.08)\}$
  ⟨*proof*⟩

theorem *g-has-limit-cycle*:
  obtains *y* **where** *g.limit-cycle y range* $(g.flow0\ y) \subseteq trapG$
⟨*proof*⟩

end

end