Lower Semicontinuous Functions

By Bogdan Grechuk

September 13, 2023

Abstract

We define the notions of lower and upper semicontinuity for functions from a metric space to the extended real line. We prove that a function is both lower and upper semicontinuous if and only if it is continuous. We also give several equivalent characterizations of lower semicontinuity. In particular, we prove that a function is lower semicontinuous if and only if its epigraph is a closed set. Also, we introduce the notion of the lower semicontinuous hull of an arbitrary function and prove its basic properties.

Contents

1 Lower semicontinuous functions 1
  1.1 Relative interior in one dimension . . . 1
  1.2 Lower and upper semicontinuity . . . 2
  1.3 Epigraphs . . . 12
  1.4 Convex Functions . . . 20

1 Lower semicontinuous functions

theory *Lower-Semicontinuous*
imports *HOL−Analysis.Multivariate-Analysis*
begin

1.1 Relative interior in one dimension

lemma *rel-interior-ereal-semiline*:
  fixes *a* :: *ereal*
  shows *rel-interior* $\{y.\ a \leq \mathit{ereal}\ y\} = \{y.\ a < \mathit{ereal}\ y\}$
proof (*cases a*)
  case (*real r*) **then show** *?thesis*
    using *rel-interior-real-semiline*[*of r*]
    by (*simp add*: *atLeast-def greaterThan-def*)
next case *PInf* **thus** *?thesis* **using** *rel-interior-empty* **by** *auto*
next case *MInf* **thus** *?thesis* **using** *rel-interior-UNIV* **by** *auto*
qed

lemma *closed-ereal-semiline*:
fixes a :: *ereal*
shows *closed* $\{y.\ a \leq$ *ereal* $y\}$
proof (*cases* a)
case (*real* r) **then show** *?thesis*
using *closed-real-atLeast* **unfolding** *atLeast-def* **by** *simp*
qed *auto*

lemma *ereal-semiline-unique*:
fixes $a\ b$ :: *ereal*
shows $\{y.\ a \leq$ *ereal* $y\} = \{y.\ b \leq$ *ereal* $y\} \longleftrightarrow a = b$
by (*metis mem-Collect-eq ereal-le-real order-antisym*)

1.2 Lower and upper semicontinuity

definition
lsc-at :: $'a \Rightarrow ('a$::*topological-space* $\Rightarrow 'b$::*order-topology*$) \Rightarrow$ *bool* **where**
lsc-at $x0\ f \longleftrightarrow (\forall X\ l.\ X \longrightarrow x0 \wedge (f \circ X) \longrightarrow l \longrightarrow f\ x0 \leq l)$

definition
usc-at :: $'a \Rightarrow ('a$::*topological-space* $\Rightarrow 'b$::*order-topology*$) \Rightarrow$ *bool* **where**
usc-at $x0\ f \longleftrightarrow (\forall X\ l.\ X \longrightarrow x0 \wedge (f \circ X) \longrightarrow l \longrightarrow l \leq f\ x0)$

lemma *lsc-at-mem*:
assumes *lsc-at* $x0\ f$
assumes $x \longrightarrow x0$
assumes $(f \circ x) \longrightarrow A$
shows $f\ x0 \leq A$
using *assms lsc-at-def*[*of* $x0\ f$] **by** *blast*

lemma *usc-at-mem*:
assumes *usc-at* $x0\ f$
assumes $x \longrightarrow x0$
assumes $(f \circ x) \longrightarrow A$
shows $f\ x0 \geq A$
using *assms usc-at-def*[*of* $x0\ f$] **by** *blast*

lemma *lsc-at-open*:
fixes f :: $'a$::*first-countable-topology* $\Rightarrow 'b$::{*complete-linorder*, *linorder-topology*}
shows *lsc-at* $x0\ f \longleftrightarrow$
$(\forall S.$ *open* $S \wedge f\ x0 \in S \longrightarrow (\exists T.$ *open* $T \wedge x0 \in T \wedge (\forall x' \in T.\ f\ x' \leq f\ x0$
$\longrightarrow f\ x' \in S)))$
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−
{ **assume** ~*?rhs*
from *this* **obtain** S **where** *S-def*:
open $S \wedge f\ x0 : S \wedge (\forall T.$ (*open* $T \wedge x0 \in T) \longrightarrow (\exists x' \in T.\ f\ x' \leq f\ x0 \wedge f\ x'$

$\notin S$)) **by** *metis*
define X **where** $X = \{x'. f\,x' \leq f\,x0 \wedge f\,x' \notin S\}$
hence $x0$ *islimpt* X **unfolding** *islimpt-def* **using** *S-def* **by** *auto*
from *this* **obtain** x **where** *x-def*: $(\forall\, n.\ x\ n \in X) \wedge x \longrightarrow x0$
using *islimpt-sequential*[*of x0 X*] **by** *auto*
hence *not*: $\sim(f \circ x) \longrightarrow (f\,x0)$ **unfolding** *lim-explicit* **using** *X-def S-def* **by** *auto*
from *compact-complete-linorder*[*of* $f \circ x$] **obtain** $l\ r$ **where** *r-def*: *strict-mono* $r \wedge ((f \circ x) \circ r) \longrightarrow l$ **by** *auto*
{ **assume** $l : S$ **hence** $\exists\, N.\ \forall\, n{\geq}N.\ f(x(r\ n)) \in S$
using *r-def lim-explicit*[*of* $f \circ x \circ r\ l$] *S-def* **by** *auto*
hence *False* **using** *x-def X-def* **by** *auto*
} **hence** *l-prop*: $l \notin S \wedge l{\leq}f\,x0$
using *r-def x-def X-def Lim-bounded*[*of* $f \circ x \circ r$]
by *auto*
{ **assume** $f\,x0 \leq l$ **hence** $f\,x0 = l$ **using** *l-prop* **by** *auto*
hence *False* **using** *l-prop S-def* **by** *auto*
}
hence $\exists\, x\ l.\ x \longrightarrow x0 \wedge (f \circ x) \longrightarrow l \wedge \sim(f\,x0 \leq l)$
apply(*rule-tac* $x{=}x \circ r$ **in** *exI*) **apply**(*rule-tac* $x{=}l$ **in** *exI*)
using *r-def x-def* **by** (*auto simp add*: *o-assoc LIMSEQ-subseq-LIMSEQ*)
hence $\sim$ *?lhs* **unfolding** *lsc-at-def* **by** *blast*
}
moreover
{ **assume** *?rhs*
{ **fix** $x\ A$ **assume** *x-def*: $x \longrightarrow x0\ (f \circ x) \longrightarrow A$
{ **assume** $A \neq f\,x0$
from *this* **obtain** $S\ V$ **where** *SV-def*: *open* $S \wedge$ *open* $V \wedge f\,x0 : S \wedge A : V \wedge S$ *Int* $V = \{\}$
using *hausdorff*[*of* $f\,x0\ A$] **by** *auto*
from *this* **obtain** T **where** *T-def*: *open* $T \wedge x0 : T \wedge (\forall\, x'{\in}T.\ (f\,x' \leq f\,x0 \longrightarrow f\,x' \in S))$
using ‹*?rhs*› **by** *metis*
from *this* **obtain** *N1* **where** $\forall\, n{\geq}N1.\ x\ n \in T$ **using** *x-def lim-explicit*[*of x x0*] **by** *auto*
hence $*$: $\forall\, n{\geq}N1.\ (f\ (x\ n) \leq f\,x0 \longrightarrow f(x\ n) \in S)$ **using** *T-def* **by** *auto*
from *SV-def* **obtain** *N2* **where** $\forall\, n{\geq}N2.\ f(x\ n) \in V$
using *lim-explicit*[*of* $f \circ x\ A$] *x-def* **by** *auto*
hence $\forall\, n{\geq}(max\ N1\ N2).\ \neg(f(x\ n) \leq f\,x0)$ **using** *SV-def* $*$ **by** *auto*
hence $\forall\, n{\geq}(max\ N1\ N2).\ f(x\ n) \geq f\,x0$ **by** *auto*
hence $f\,x0 \leq A$ **using** *Lim-bounded2*[*of* $f \circ x\ A$ *max N1 N2* $f\,x0$] *x-def* **by** *auto*
} **hence** $f\,x0 \leq A$ **by** *auto*
} **hence** *?lhs* **unfolding** *lsc-at-def* **by** *blast*
} **ultimately show** *?thesis* **by** *blast*
qed

lemma *lsc-at-open-mem*:

fixes f :: $'a$::*first-countable-topology* $\Rightarrow$ $'b$::{*complete-linorder*, *linorder-topology*}
assumes *lsc-at x0 f*
assumes *open S* $\wedge$ *f x0* : *S*
obtains *T* **where** *open T* $\wedge$ *x0* $\in$ *T* $\wedge$ ($\forall x'\in T$. (*f x'* $\leq$ *f x0* $\longrightarrow$ *f x'* $\in$ *S*))
using *assms lsc-at-open*[*of x0 f*] **by** *blast*

lemma *lsc-at-MInfty*:
fixes f :: $'a$::*topological-space* $\Rightarrow$ *ereal*
assumes *f x0* = $-\infty$
shows *lsc-at x0 f*
unfolding *lsc-at-def* **using** *assms* **by** *auto*

lemma *lsc-at-PInfty*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
assumes *f x0* = ∞
shows *lsc-at x0 f* $\longleftrightarrow$ *continuous* (*at x0*) *f*
unfolding *lsc-at-open continuous-at-open* **using** *assms* **by** *auto*

lemma *lsc-at-real*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
assumes |*f x0*| $\neq \infty$
shows *lsc-at x0 f* $\longleftrightarrow$ ($\forall$ *e*. *e*>*0* $\longrightarrow$ ($\exists$ *T*. *open T* $\wedge$ *x0* : *T* $\wedge$ ($\forall$ *y* $\in$ *T*. *f y* > *f x0* − *e*)))
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−
obtain *m* **where** *m-def*: *f x0* = *ereal m* **using** *assms* **by** (*cases f x0*) *auto*
{ assume *lsc*: *lsc-at x0 f*
{ fix *e* **assume** *e-def*: (*e* :: *ereal*)>*0*
hence ∗: *f x0* : {*f x0* − *e* <..< *f x0* + *e*} **using** *assms ereal-between* **by** *auto*
from *this* **obtain** *T* **where** *T-def*: *open T* $\wedge$ *x0* : *T* $\wedge$ ($\forall x'\in T$. *f x'* $\leq$ *f x0* $\longrightarrow$ *f x'* $\in$ {*f x0* − *e* <..< *f x0* + *e*})
apply (*subst lsc-at-open-mem*[*of x0 f* {*f x0* − *e* <..< *f x0* + *e*}]) **using** *lsc e-def* **by** *auto*
{ fix *y* **assume** *y*:*T*
{ assume *f y* $\leq$ *f x0* **hence** *f y* > *f x0* − *e* **using** *T-def* ‹*y*:*T*› **by** *auto* **}**
moreover
{ assume *f y* > *f x0* **hence** ...>*f x0* − *e* **using** ∗ **by** *auto* **}**
ultimately have *f y* > *f x0* − *e* **using** *not-le* **by** *blast*
} hence $\exists$ *T*. *open T* $\wedge$ *x0* $\in$ *T* $\wedge$ ($\forall$ *y* $\in$ *T*. *f y* > *f x0* − *e*) **using** *T-def* **by** *auto*
} hence *?rhs* **by** *auto*
}
moreover
{ assume *?rhs*
{ fix *S* **assume** *S-def*: *open S* $\wedge$ *f x0* : *S*
from *this* **obtain** *e* **where** *e-def*: *e*>*0* $\wedge$ {*f x0* − *e*<..<*f x0* + *e*} $\leq$ *S*

apply (*subst ereal-open-cont-interval*[*of S f x0*]) **using** *assms* **by** *auto*
from *this* **obtain** *T* **where** *T-def*: *open T* ∧ *x0* : *T* ∧ (∀ *y* ∈ *T*. *f y* > *f x0* − *e*)
using ‹*?rhs*›[*rule-format*, *of e*] **by** *auto*
{ **fix** *y* **assume** *y*:*T f y* ≤ *f x0* **hence** *f y* < *f x0* + *e*
using *assms e-def ereal-between*[*of f x0 e*] **by** *auto*
hence *f y* ∈ *S* **using** *e-def T-def* ‹*y*∈*T*› **by** *auto*
} **hence** ∃ *T*. *open T* ∧ *x0* : *T* ∧ (∀ *y*∈*T*. (*f y* ≤ *f x0* ⟶ *f y* ∈ *S*)) **using** *T-def* **by** *auto*
} **hence** *lsc-at x0 f* **using** *lsc-at-open* **by** *auto*
} **ultimately show** *?thesis* **by** *auto*
qed

lemma *lsc-at-ereal*:
fixes *f* :: *'a*::*metric-space* ⇒ *ereal*
shows *lsc-at x0 f* ⟷ (∀ *C*<*f*(*x0*). ∃ *T*. *open T* ∧ *x0* ∈ *T* ∧ (∀ *y* ∈ *T*. *f y* > *C*))
(**is** *?lhs* ⟷ *?rhs*)
proof−
{ **assume** *f x0* = −∞ **hence** *?thesis* **using** *lsc-at-MInfty* **by** *auto* }
moreover
{ **assume** *pinf*: *f x0* = ∞
{ **assume** *lsc*: *lsc-at x0 f*
{ **fix** *C* **assume** *C*<*f x0*
hence *open* {*C*<..} ∧ *f x0* : {*C*<..} **by** *auto*
from *this* **obtain** *T* **where** *T-def*: *open T* ∧ *x0* ∈ *T* ∧ (∀ *y*∈*T*. *f y* ∈ {*C*<..})
using *pinf lsc lsc-at-PInfty*[*of f x0*] **unfolding** *continuous-at-open* **by** *metis*
hence ∃ *T*. *open T* ∧ *x0* ∈ *T* ∧ (∀ *y*∈*T*. *C* < *f y*) **by** *auto*
} **hence** *?rhs* **by** *auto*
}
moreover
{ **assume** *?rhs*
{ **fix** *S* **assume** *S-def*: *open S* ∧ *f x0* : *S*
then obtain *C* **where** *C-def*: *ereal C* < *f x0* ∧ {*ereal C*<..} ≤ *S* **using** *pinf open-PInfty* **by** *auto*
then obtain *T* **where** *T-def*: *open T* ∧ *x0* : *T* ∧ (∀ *y* ∈ *T*. *f y* ∈ *S*)
using ‹*?rhs*›[*rule-format*, *of ereal C*] **by** *auto*
hence ∃ *T*. *open T* ∧ *x0* ∈ *T* ∧ (∀ *y*∈*T*. (*f y* ≤ *f x0* ⟶ *f y* ∈ *S*)) **using** *T-def* **by** *auto*
} **hence** *lsc-at x0 f* **using** *lsc-at-open* **by** *auto*
} **ultimately have** *?thesis* **by** *blast*
}
moreover
{ **assume** *fin*: *f x0* ≠ −∞ *f x0* ≠ ∞
{ **assume** *lsc*: *lsc-at x0 f*
{ **fix** *C* **assume** *C*<*f x0*
hence *f x0*−*C*>*0* **using** *fin ereal-less-minus-iff* **by** *auto*
from *this* **obtain** *T* **where** *T-def*: *open T* ∧ *x0* ∈ *T* ∧ (∀ *y*∈*T*. *f x0* − (*f x0*−*C*) < *f y*)

using *lsc-at-real*[*of f x0*] *lsc fin* **by** *auto*
moreover have $f\ x0\ -\ (f\ x0 - C)\ =\ C$ **using** *fin* **apply** (*cases f x0*, *cases C*) **by** *auto*
ultimately have $\exists\, T.\ open\ T \wedge x0 \in T \wedge (\forall\, y{\in}T.\ C < f\ y)$ **by** *auto*
} **hence** *?rhs* **by** *auto*
}
moreover
{ **assume** *?rhs*
{ **fix** e :: *ereal* **assume** $e{>}0$
hence $f\ x0\ -\ e < f\ x0$ **using** *fin* **apply** (*cases f x0*, *cases e*) **by** *auto*
hence $\exists\, T.\ open\ T \wedge x0 \in T \wedge (\forall\, y{\in}T.\ f\ x0\ -\ e < f\ y)$ **using** *fin* ‹*?rhs*› **by** *auto*
} **hence** *lsc-at x0 f* **using** *lsc-at-real*[*of f x0*] *fin* **by** *auto*
} **ultimately have** *?thesis* **by** *blast*
} **ultimately show** *?thesis* **by** *blast*
qed

lemma *lst-at-ball*:
fixes f :: *'a::metric-space => ereal*
shows *lsc-at x0 f* $\longleftrightarrow$ $(\forall\, C{<}f(x0).\ \exists\, d{>}0.\ \forall\, y \in (ball\ x0\ d).\ C{<}f(y))$
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof
assume *lsc*: *lsc-at x0 f*
show *?rhs*
proof (*intro strip*)
fix C :: *ereal* **assume** $C{<}f\ x0$
then obtain T **where** $open\ T \wedge x0 : T \wedge (\forall\, y \in T.\ C < f\ y)$
using *lsc lsc-at-ereal*[*of x0 f*] **by** *auto*
then show $\exists\, d.\ d{>}0 \wedge (\forall\, y \in (ball\ x0\ d).\ C < f\ y)$
by (*force simp add*: *open-contains-ball*)
qed

next
assume *?rhs*
{ **fix** C :: *ereal* **assume** $C{<}f\ x0$
then obtain d **where** $d{>}0 \wedge (\forall\, y \in (ball\ x0\ d).\ C < f\ y)$ **using** ‹*?rhs*› **by** *auto*
hence $\exists\, T.\ open\ T \wedge x0 \in T \wedge (\forall\, y \in T.\ C < f\ y)$
by (*meson Elementary-Metric-Spaces.open-ball centre-in-ball*)
} **then show** *?lhs* **using** *lsc-at-ereal*[*of x0 f*] **by** *auto*
qed

lemma *lst-at-delta*:
fixes f :: *'a::metric-space* $\Rightarrow$ *ereal*
shows *lsc-at x0 f* $\longleftrightarrow$ $(\forall\, C{<}f(x0).\ \exists\, d{>}0.\ \forall\, y.\ dist\ x0\ y < d \longrightarrow C < f\ y)$
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−

```
  have ?rhs ⟷ (∀ C<f(x0). ∃ d>0. ∀ y ∈ (ball x0 d). C < f y) unfolding ball-def
by auto
  thus ?thesis using lst-at-ball[of x0 f] by auto
qed


lemma lsc-liminf-at:
  fixes f :: 'a::metric-space ⇒ ereal
  shows lsc-at x0 f ⟷ f x0 ≤ Liminf (at x0) f
  unfolding lst-at-ball le-Liminf-iff eventually-at
  by (intro arg-cong[where f=All] imp-cong refl ext ex-cong)
     (auto simp: dist-commute zero-less-dist-iff)


lemma lsc-liminf-at-eq:
  fixes f :: 'a::metric-space ⇒ ereal
  shows lsc-at x0 f ⟷ (f x0 = min (f x0) (Liminf (at x0) f))
by (metis inf-ereal-def le-iff-inf lsc-liminf-at)


lemma lsc-imp-liminf:
fixes f :: 'a::metric-space ⇒ ereal
assumes lsc-at x0 f
assumes x ⟶ x0
shows f x0 ≤ liminf (f ∘ x)
proof (cases f x0)
 case PInf then show ?thesis using assms lsc-at-PInfty[of f x0] lim-imp-Liminf[of
- f ∘ x]
    continuous-at-sequentially[of x0 f] by auto
next
  case (real r)
  { fix e assume e-def: (e :: ereal)>0
    from this obtain T where T-def: open T ∧ x0 : T ∧ (∀ y ∈ T. f y > f x0 −
e)
       using lsc-at-real[of f x0] real assms by auto
    from this obtain N where N-def: ∀ n≥N. x n ∈ T
       apply (subst tendsto-obtains-N[of x x0 T]) using assms by auto
    hence ∀ n≥N. f x0 − e < (f ∘ x) n using T-def by auto
     hence liminf (f ∘ x) ≥ f x0 − e by (intro Liminf-bounded) (auto simp:
eventually-sequentially intro!: exI[of - N])
    hence f x0 ≤ liminf (f ∘ x) + e apply (cases e) unfolding ereal-minus-le-iff
by auto
  }
  then show ?thesis
    using ereal-le-epsilon by blast
qed auto

lemma lsc-liminf:
  fixes f :: 'a::metric-space ⇒ ereal
```

```
  shows lsc-at x0 f ⟷ (∀ x. x ⟶ x0 ⟶ f x0 ≤ liminf (f ∘ x))
    (is ?lhs ⟷ ?rhs)
proof
  assume ?rhs
    { fix x A assume x-def: x ⟶ x0 (f ∘ x) ⟶ A
      hence f x0 ≤ A using ‹?rhs› lim-imp-Liminf[of sequentially] by auto
    } thus ?lhs unfolding lsc-at-def by blast
qed (use lsc-imp-liminf in auto)


lemma lsc-sequentially:
  fixes f :: 'a::metric-space ⇒ ereal
  shows lsc-at x0 f ⟷ (∀ x c. x ⟶ x0 ∧ (∀ n. f(x n)≤c) ⟶ f(x0)≤c)
    (is ?lhs ⟷ ?rhs)
proof
  assume ?rhs
  { fix x l assume x ⟶ x0 (f ∘ x) ⟶ l
    { assume l = ∞ hence f x0 ≤ l by auto }
    moreover
    { assume l = −∞
      { fix B :: real obtain N where N-def: ∀ n≥N. f(x n) ≤ ereal B
          using Lim-MInfty[of f ∘ x] ‹(f ∘ x) ⟶ l› ‹l = −∞› by auto
        define g where g n = (if n≥N then x n else x N) for n
        hence g ⟶ x0
          by (intro filterlim-cong[THEN iffD1, OF refl refl - ‹x ⟶ x0›])
            (auto simp: eventually-sequentially)
        moreover have ∀ n. f(g n) ≤ ereal B using g-def N-def by auto
        ultimately have f x0 ≤ ereal B using ‹?rhs› by auto
      } hence f x0 = −∞ using ereal-bot by auto
      hence f x0 ≤ l by auto }
    moreover
    { assume fin: |l| ≠ ∞
      { fix e assume e-def: (e :: ereal)>0
        from this obtain N where N-def: ∀ n≥N. f(x n) ∈ {l − e <..< l + e}
          apply (subst tendsto-obtains-N[of f ∘ x l {l − e <..< l + e}])
          using fin e-def ereal-between ‹(f ∘ x) ⟶ l› by auto
        define g where g n = (if n≥N then x n else x N) for n
        hence g ⟶ x0
          by (intro filterlim-cong[THEN iffD1, OF refl refl - ‹x ⟶ x0›])
            (auto simp: eventually-sequentially)
        moreover have ∀ n. f(g n) ≤ l + e using g-def N-def by auto
        ultimately have f x0 ≤ l + e using ‹?rhs› by auto
      } hence f x0 ≤ l using ereal-le-epsilon by auto
    } ultimately have f x0 ≤ l by blast
  } then show ?lhs unfolding lsc-at-def by auto
next
  assume lsc: ?lhs
  { fix x c assume xc-def: x ⟶ x0 ∧ (∀ n. f(x n)≤c)
    hence liminf (f ∘ x) ≤ c
```

using *Limsup-bounded*[**where** F = *sequentially* **and** $X = f \circ x$ **and** $C = c$]
Liminf-le-Limsup[*of sequentially* $f \circ x$]
by *auto*
hence $f\ x0 \le c$ **using** *lsc xc-def lsc-imp-liminf*[*of x0 f x*] **by** *auto*
} **thus** *?rhs* **by** *auto*
qed

lemma *lsc-sequentially-gen*:
fixes f :: *'a::metric-space* ⇒ *ereal*
shows *lsc-at x0 f* ⟷ (∀ *x c c0*. $x \longrightarrow x0 \wedge c \longrightarrow c0 \wedge (\forall n.\ f(x\ n) \le c\ n) \longrightarrow f(x0) \le c0$)
(**is** *?lhs* ⟷ *?rhs*)
proof
assume *?rhs*
{ **fix** *x c0* **assume** *a*: $x \longrightarrow x0 \wedge (\forall n.\ f\ (x\ n) \le c0)$
define *c* **where** $c = (\lambda n::nat.\ c0)$
hence $c \longrightarrow c0$ **by** *auto*
hence $f(x0) \le c0$ **using** ‹*?rhs*›[*rule-format, of x c c0*] **using** *a c-def* **by** *auto*
} **then show** *?lhs* **using** *lsc-sequentially*[*of x0 f*] **by** *auto*
next
assume *lsc*: *lsc-at x0 f*
{ **fix** *x c c0* **assume** *xc-def*: $x \longrightarrow x0 \wedge c \longrightarrow c0 \wedge (\forall n.\ f(x\ n) \le c\ n)$
hence *liminf* $(f \circ x) \le c0$
using *Liminf-mono*[*of* $f \circ x$ *c sequentially*] *lim-imp-Liminf*[*of sequentially*] **by** *auto*
hence $f\ x0 \le c0$ **using** *lsc xc-def lsc-imp-liminf*[*of x0 f x*] **by** *auto*
} **then show** *?rhs* **by** *auto*
qed

lemma *lsc-sequentially-mem*:
fixes f :: *'a::metric-space* ⇒ *ereal*
assumes *lsc-at x0 f*
assumes $x \longrightarrow x0\ c \longrightarrow c0$
assumes $\forall n.\ f(x\ n) \le c\ n$
shows $f(x0) \le c0$
using *lsc-sequentially-gen*[*of x0 f*] *assms* **by** *auto*

lemma *lsc-uminus*:
fixes f :: *'a::metric-space* ⇒ *ereal*
shows *lsc-at x0* $(\lambda x.\ -f\ x)$ ⟷ *usc-at x0 f*
proof
assume *lsc*: *lsc-at x0* $(\lambda x.\ -f\ x)$
{ **fix** *x A* **assume** *x-def*: $x \longrightarrow x0\ (f \circ x) \longrightarrow A$
hence $(\lambda i.\ -\ f\ (x\ i)) \longrightarrow -A$ **using** *tendsto-uminus-ereal*[*of* $f \circ x$ *A*] **by** *auto*
hence $((\lambda x.\ -\ f\ x) \circ x) \longrightarrow -A$ **unfolding** *o-def* **by** *auto*

hence $-f\,x0 \le -A$ **apply** (*subst lsc-at-mem*[*of x0* ($\lambda x.\ -f\ x$) *x*]) **using** *lsc x-def* **by** *auto*
hence $f\ x0 \ge A$ **by** *auto*
} **then show** *usc-at x0 f* **unfolding** *usc-at-def* **by** *auto*

next
assume *usc*: *usc-at x0 f*
{ **fix** *x A* **assume** *x-def*: $x \longrightarrow x0$ (($\lambda x.\ -f\ x) \circ x) \longrightarrow A$
hence ($\lambda i.\ -f\ (x\ i)) \longrightarrow A$ **unfolding** *o-def* **by** *auto*
hence ($\lambda i.\ f\ (x\ i)) \longrightarrow -A$ **using** *tendsto-uminus-ereal*[*of* ($\lambda i.\ -f\ (x\ i)$) *A*] **by** *auto*
hence $(f \circ x) \longrightarrow -A$ **unfolding** *o-def* **by** *auto*
hence $f\ x0 \ge -A$ **apply** (*subst usc-at-mem*[*of x0 f x*]) **using** *usc x-def* **by** *auto*
hence $-f\ x0 \le A$ **by** (*auto simp*: *ereal-uminus-le-reorder*)
} **then show** *lsc-at x0* ($\lambda x.\ -f\ x$) **unfolding** *lsc-at-def* **by** *auto*
qed

lemma *usc-limsup*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
shows *usc-at x0 f* $\longleftrightarrow$ ($\forall x.\ x \longrightarrow x0 \longrightarrow f\ x0 \ge$ *limsup* $(f \circ x)$)
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−
have *usc-at x0 f* $\longleftrightarrow$ ($\forall x.\ x \longrightarrow x0 \longrightarrow -f\ x0 \le$ *liminf* (($\lambda x.\ -f\ x) \circ x$))
using *lsc-uminus*[*of x0 f*] *lsc-liminf*[*of x0* ($\lambda x.\ -f\ x$)] **by** *auto*
moreover
{ **fix** x **assume** $x \longrightarrow x0$
have ($-f\ x0 \le -$*limsup* $(f \circ x)) \longleftrightarrow (-f\ x0 \le$ *liminf* (($\lambda x.\ -f\ x) \circ x$))
using *ereal-Liminf-uminus*[*of* - $f \circ x$] **unfolding** *o-def* **by** *auto*
hence ($f\ x0 \ge$ *limsup* $(f \circ x)) \longleftrightarrow (-f\ x0 \le$ *liminf* (($\lambda x.\ -f\ x) \circ x$))
by *auto*
} **ultimately show** *?thesis* **by** *auto*
qed

lemma *usc-imp-limsup*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
assumes *usc-at x0 f*
assumes $x \longrightarrow x0$
shows $f\ x0 \ge$ *limsup* $(f \circ x)$
using *assms usc-limsup*[*of x0 f*] **by** *auto*

lemma *usc-limsup-at*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
shows *usc-at x0 f* $\longleftrightarrow f\ x0 \ge$ *Limsup* (*at x0*) *f*
proof−
have *usc-at x0 f* $\longleftrightarrow$ *lsc-at x0* ($\lambda x.\ -(f\ x)$) **by** (*metis lsc-uminus*)

also have ... $\longleftrightarrow -(f\,x0) \leq$ *Liminf* (*at x0*) $(\lambda x. -(f\,x))$ **by** (*metis lsc-liminf-at*)
also have ... $\longleftrightarrow -(f\,x0) \leq -($*Limsup* (*at x0*) $f)$ **by** (*metis ereal-Liminf-uminus*)
finally show *?thesis* **by** *auto*
qed

lemma *continuous-iff-lsc-usc*:
fixes f :: *'a::metric-space* $\Rightarrow$ *ereal*
shows *continuous* (*at x0*) $f \longleftrightarrow$ (*lsc-at x0 f*) $\wedge$ (*usc-at x0 f*)
proof−
{ **assume** *a*: *continuous* (*at x0*) *f*
{ **fix** *x* **assume** $x \longrightarrow x0$
hence $(f \circ x) \longrightarrow f\,x0$ **using** *a continuous-imp-tendsto*[*of x0 f x*] **by** *auto*
hence *liminf* $(f \circ x) = f\,x0 \wedge$ *limsup* $(f \circ x) = f\,x0$
using *lim-imp-Liminf*[*of sequentially*] *lim-imp-Limsup*[*of sequentially*] **by** *auto*
} **hence** *lsc-at x0 f* $\wedge$ *usc-at x0 f* **unfolding** *lsc-liminf usc-limsup* **by** *auto*
}
moreover
{ **assume** *a*: (*lsc-at x0 f*) $\wedge$ (*usc-at x0 f*)
{ **fix** *x* **assume** $x \longrightarrow x0$
hence *limsup* $(f \circ x) \leq f\,x0$ **using** *a* **unfolding** *usc-limsup* **by** *auto*
moreover have ... $\leq$ *liminf* $(f \circ x)$ **using** *a* ‹$x \longrightarrow x0$› **unfolding** *lsc-liminf* **by** *auto*
ultimately have *limsup* $(f \circ x) = f\,x0 \wedge$ *liminf* $(f \circ x) = f\,x0$
using *Liminf-le-Limsup*[*of sequentially* $f \circ x$] **by** *auto*
hence $(f \circ x) \longrightarrow f\,x0$ **using** *Liminf-eq-Limsup*[*of sequentially*]
by (*simp add*: *tendsto-iff-Liminf-eq-Limsup*)
} **hence** *continuous* (*at x0*) *f*
using *continuous-at-sequentially*[*of x0 f*] **by** *auto*
} **ultimately show** *?thesis* **by** *blast*
qed

lemma *continuous-lsc-compose*:
assumes *lsc-at* (*g x0*) *f continuous* (*at x0*) *g*
shows *lsc-at x0* $(f \circ g)$
proof−
{ **fix** *x L* **assume** $x \longrightarrow x0$ $(f \circ g \circ x) \longrightarrow L$
hence $f(g\,x0) \leq L$ **apply** (*subst lsc-at-mem*[*of g x0 f g* $\circ$ *x L*])
using *assms continuous-imp-tendsto*[*of x0 g x*] **unfolding** *o-def* **by** *auto*
} **from** *this* **show** *?thesis* **unfolding** *lsc-at-def* **by** *auto*
qed

lemma *continuous-isCont*:
continuous (*at x0*) $f \longleftrightarrow$ *isCont f x0*
by (*metis continuous-at isCont-def*)

lemma *isCont-iff-lsc-usc*:
  fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
  shows *isCont* f *x0* $\longleftrightarrow$ (*lsc-at x0 f*) $\wedge$ (*usc-at x0 f*)
by (*metis continuous-iff-lsc-usc continuous-isCont*)

definition
  lsc :: ($'a$::*topological-space* $\Rightarrow$ $'b$::*order-topology*) $\Rightarrow$ *bool* **where**
  lsc f $\longleftrightarrow$ ($\forall x$. *lsc-at x f*)

definition
  usc :: ($'a$::*topological-space* $\Rightarrow$ $'b$::*order-topology*) $\Rightarrow$ *bool* **where**
  usc f $\longleftrightarrow$ ($\forall x$. *usc-at x f*)

lemma *continuous-UNIV-iff-lsc-usc*:
  fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
  shows ($\forall x$. *continuous* (*at x*) *f*) $\longleftrightarrow$ (*lsc f*) $\wedge$ (*usc f*)
by (*metis continuous-iff-lsc-usc lsc-def usc-def*)

1.3 Epigraphs

definition *Epigraph S* (*f*::- $\Rightarrow$ *ereal*) = $\{xy.$ *fst xy* : S $\wedge$ f (*fst xy*) $\leq$ *ereal*(*snd xy*)$\}$

lemma *mem-Epigraph*: $(x, y) \in$ *Epigraph S f* $\longleftrightarrow$ $x \in S \wedge f\, x \leq$ *ereal y* **unfolding** *Epigraph-def* **by** *auto*

lemma *ereal-closed-levels*:
fixes f :: $'a$::*metric-space* $\Rightarrow$ *ereal*
shows ($\forall y$. *closed* $\{x.\ f(x) \leq y\}$) $\longleftrightarrow$ ($\forall r$. *closed* $\{x.\ f(x) \leq$ *ereal r*$\}$)
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−
{ assume *?rhs*
  { fix y :: *ereal*
   { assume $y \neq \infty \wedge y \neq -\infty$ **hence** *closed* $\{x.\ f(x) \leq y\}$ **using** ‹*?rhs*› **by** (*cases y*) *auto* **}**
    moreover
    { assume $y = \infty$ **hence** *closed* $\{x.\ f(x) \leq y\}$ **by** *auto* **}**
    moreover
    { assume $y = -\infty$
      hence $\{x.\ f(x) \leq y\}$ = *Inter* $\{\{x.\ f(x) \leq$ *ereal r*$\}$ $|r.\ r$: *UNIV*$\}$ **using** *ereal-bot* **by** *auto*
      hence *closed* $\{x.\ f(x) \leq y\}$ **using** *closed-Inter* ‹*?rhs*› **by** *auto*
    } ultimately have *closed* $\{x.\ f(x) \leq y\}$ **by** *blast*
  } hence *?lhs* **by** *auto*

} **from** *this* **show** *?thesis* **by** *auto*
qed

lemma *lsc-iff*:
fixes *f* :: *'a::metric-space* $\Rightarrow$ *ereal*
shows (*lsc f* $\longleftrightarrow$ ($\forall$ *y*. *closed* {*x*. *f*(*x*)$\leq$*y*})) $\wedge$ (*lsc f* $\longleftrightarrow$ *closed* (*Epigraph UNIV f*))
proof−
{ **assume** *lsc f*
{ **fix** *z z0* **assume** *a*: $\forall$ *n*. *z n* $\in$ (*Epigraph UNIV f*) $\wedge$ *z* $\longrightarrow$ *z0*
{ **fix** *n* **have** *z n* : (*Epigraph UNIV f*) **using** *a* **by** *auto*
hence *f* (*fst* (*z n*)) $\leq$ *ereal*(*snd* (*z n*)) **using** *a* **unfolding** *Epigraph-def* **by** *auto*
hence $\exists$ *xn cn*. *z n* = (*xn*, *cn*) $\wedge$ *f*(*xn*)$\leq$*ereal cn*
apply (*rule-tac x=fst* (*z n*) **in** *exI*) **apply** (*rule-tac x=snd* (*z n*) **in** *exI*) **by** *auto*
} **from** *this* **obtain** *x c* **where** *xc-def*: $\forall$ *n*. *z n* = (*x n*, *c n*) $\wedge$ *f*(*x n*)$\leq$*ereal* (*c n*) **by** *metis*
from *this a* **have** $\exists$ *x0 c0*. *z0* = (*x0*, *c0*) $\wedge$ *x* $\longrightarrow$ *x0* $\wedge$ *c* $\longrightarrow$ *c0*
apply (*rule-tac x=fst z0* **in** *exI*) **apply** (*rule-tac x=snd z0* **in** *exI*)
using *tendsto-fst*[*of z z0*] *tendsto-snd*[*of z z0*] **by** *auto*
from *this* **obtain** *x0 c0* **where** *xc0-def*: *z0* = (*x0*, *c0*) $\wedge$ *x* $\longrightarrow$ *x0* $\wedge$ *c* $\longrightarrow$ *c0* **by** *auto*
hence *f*(*x0*)$\leq$*ereal c0* **apply** (*subst lsc-sequentially-mem*[*of x0 f x ereal* $\circ$ *c ereal c0*])
using ‹*lsc f*› *xc-def* **unfolding** *lsc-def* **unfolding** *o-def* **by** *auto*
hence *z0* : (*Epigraph UNIV f*) **unfolding** *Epigraph-def* **using** *xc0-def* **by** *auto*
} **hence** *closed* (*Epigraph UNIV f*) **by** (*simp add*: *closed-sequential-limits*)
}
moreover
{ **assume** *closed* (*Epigraph UNIV f*)
hence $*$: $\forall$ *x l*. ($\forall$ *n*. *f* (*fst* (*x n*)) $\leq$ *ereal*(*snd* (*x n*))) $\wedge$ *x* $\longrightarrow$ *l* $\longrightarrow$
f (*fst l*) $\leq$ *ereal*(*snd l*) **unfolding** *Epigraph-def closed-sequential-limits* **by** *auto*
{ **fix** *r* :: *real*
{ **fix** *z z0* **assume** *a*: $\forall$ *n*. *f*(*z n*)$\leq$*ereal r* $\wedge$ *z* $\longrightarrow$ *z0*
hence *f*(*z0*)$\leq$*ereal r* **using** $*$[*rule-format*, *of* (λ*n*. (*z n*,*r*)) (*z0*,*r*)]
tendsto-Pair[*of z z0*] **by** *auto*
} **hence** *closed* {*x*. *f*(*x*)$\leq$*ereal r*} **by** (*simp add*: *closed-sequential-limits*)
} **hence** $\forall$ *y*. *closed* {*x*. *f*(*x*)$\leq$*y*} **using** *ereal-closed-levels* **by** *auto*
}
moreover
{ **assume** *a*: $\forall$ *y*. *closed* {*x*. *f*(*x*)$\leq$ *y*}
{ **fix** *x0*
{ **fix** *x l* **assume** *x* $\longrightarrow$ *x0* (*f* $\circ$ *x*) $\longrightarrow$ *l*
{ **assume** *l* = ∞ **hence** *f x0* $\leq$ *l* **by** *auto* **}**
moreover
{ **assume** *mi*: *l* = $-\infty$
{ **fix** *B* :: *real*

```
          obtain N where N-def: ∀ n≥N. f(x n)≤ereal B
            using mi ‹(f ∘ x) ⟶ l› Lim-MInfty[of f ∘ x] by auto
          { fix d assume (d::real)>0
            from this obtain N1 where N1-def: ∀ n≥N1. dist (x n) x0 < d
               using ‹x ⟶ x0› unfolding lim-sequentially by auto
            hence ∃ y. dist y x0 < d ∧ y : {x. f(x)≤ereal B}
              apply (rule-tac x=x (max N N1) in exI) using N-def by auto
          }
          hence x0 : closure {x. f(x)≤ereal B} apply (subst closure-approachable)
by auto
          hence f x0 ≤ ereal B using a by auto
        } hence f x0 ≤ l using ereal-bot[of f x0] by auto
      }
      moreover
      { assume fin: |l| ≠ ∞
        { fix e assume e-def: (e :: ereal)>0
          from this obtain N where N-def: ∀ n≥N. f(x n) : {l − e <..< l + e}
            apply (subst tendsto-obtains-N[of f ∘ x l {l − e <..< l + e}])
            using fin e-def ereal-between ‹(f ∘ x) ⟶ l› by auto
          hence *: ∀ n≥N. x n : {x. f(x)≤l+e} using N-def by auto
          { fix d assume (d::real)>0
            from this obtain N1 where N1-def: ∀ n≥N1. dist (x n) x0 < d
               using ‹x ⟶ x0› unfolding lim-sequentially by auto
            hence ∃ y. dist y x0 < d ∧ y : {x. f(x)≤l+e}
              apply (rule-tac x=x (max N N1) in exI) using * by auto
          }
          hence x0 : closure {x. f(x)≤l+e} apply (subst closure-approachable) by
auto
          hence f x0 ≤ l+e using a by auto
        } hence f x0 ≤ l using ereal-le-epsilon by auto
      } ultimately have f x0 ≤ l by blast
    } hence lsc-at x0 f unfolding lsc-at-def by auto
  } hence lsc f unfolding lsc-def by auto
}
ultimately show ?thesis by auto
qed


definition lsc-hull :: ('a::metric-space ⇒ ereal) ⇒ ('a::metric-space ⇒ ereal) where
  lsc-hull f = (SOME g. Epigraph UNIV g = closure(Epigraph UNIV f))


lemma epigraph-mono:
  fixes f :: 'a::metric-space ⇒ ereal
  shows (x,y):Epigraph UNIV f ∧ y≤z ⟶ (x,z):Epigraph UNIV f
unfolding Epigraph-def apply auto
using ereal-less-eq(3)[of y z] order-trans-rules(23) by blast
```

lemma *closed-epigraph-lines*:
fixes *S* :: (*'a*::*metric-space* ∗ *'b*::*metric-space*) *set*
assumes *closed S*
shows *closed* {*z*. (*x*, *z*) : *S*}
proof−
{ **fix** *z* **assume** *z-def*: *z* : *closure* {*z*. (*x*, *z*) : *S*}
  { **fix** *e* :: *real* **assume** *e*>*0*
    from *this* **obtain** *y* **where** *y-def*: (*x*,*y*) : *S* ∧ *dist y z* < *e*
      using *closure-approachable*[*of z* {*z*. (*x*, *z*) : *S*}] *z-def* **by** *auto*
    moreover have *dist* (*x*,*y*) (*x*,*z*) = *dist y z* **unfolding** *dist-prod-def* **by** *auto*
    ultimately have ∃ *s*. *s* ∈ *S* ∧ *dist s* (*x*,*z*) < *e* **apply**(*rule-tac x*=(*x*,*y*) **in** *exI*)
by *auto*
  } **hence** (*x*,*z*):*S* **using** *closed-approachable*[*of S* (*x*,*z*)] *assms* **by** *auto*
} **hence** *closure* {*z*. (*x*, *z*) : *S*} ≤ {*z*. (*x*, *z*) : *S*} **by** *blast*
from *this* **show** *?thesis* **using** *closure-subset-eq* **by** *auto*
qed

lemma *mono-epigraph*:
  fixes *S* :: (*'a*::*metric-space* ∗ *real*) *set*
  assumes *mono*: ∀ *x y z*. (*x*,*y*):*S* ∧ *y*≤*z* ⟶ (*x*,*z*):*S*
  assumes *closed S*
  shows ∃ *g*. ((*Epigraph UNIV g*) = *S*)
proof−
{ **fix** *x*
  have *closed* {*z*. (*x*, *z*) : *S*} **using** ‹*closed S*› *closed-epigraph-lines* **by** *auto*
  hence ∃ *a*. {*z*. (*x*, *z*) : *S*} = {*z*. *a* ≤ *ereal z*} **apply** (*subst mono-closed-ereal*)
using *mono* **by** *auto*
} **from** *this* **obtain** *g* **where** *g-def*: ∀ *x*. {*z*. (*x*, *z*) : *S*} = {*z*. *g x* ≤ *ereal z*} **by**
metis
{ **fix** *s*
  have *s*:*S* ⟷ (*fst s*, *snd s*) : *S* **by** *auto*
  also have ... ⟷ *g*(*fst s*) ≤ *ereal* (*snd s*) **using** *g-def*[*rule-format*, *of fst s*] **by**
blast
  finally have *s*:*S* ⟷ *g*(*fst s*) ≤ *ereal* (*snd s*) **by** *auto*
}
hence (*Epigraph UNIV g*) = *S* **unfolding** *Epigraph-def* **by** *auto*
from *this* **show** *?thesis* **by** *auto*
qed

lemma *lsc-hull-exists*:
  fixes *f* :: *'a*::*metric-space* ⇒ *ereal*
  shows ∃ *g*. *Epigraph UNIV g* = *closure* (*Epigraph UNIV f*)
proof−
{ **fix** *x y z* **assume** *xy*: (*x*,*y*):*closure* (*Epigraph UNIV f*) ∧ *y*≤*z*
  { **fix** *e* :: *real* **assume** *e*>*0*
    hence ∃ *ya*∈*Epigraph UNIV f*. *dist ya* (*x*, *y*) < *e*

using *xy closure-approachable*[*of* (*x*,*y*) *Epigraph UNIV f*] **by** *auto*
from *this* **obtain** *a b* **where** *ab*: (*a*,*b*):*Epigraph UNIV f* ∧ *dist* (*a*,*b*) (*x*,*y*) < *e* **by** *auto*
moreover have *dist* (*a*,*b*) (*x*,*y*) = *dist* (*a*,*b*+(*z*−*y*)) (*x*,*z*)
unfolding *dist-prod-def dist-norm* **by** (*simp add*: *algebra-simps*)
moreover have (*a*,*b*+(*z*−*y*)):*Epigraph UNIV f* **apply** (*subst epigraph-mono*[*of* - *b*]) **using** *ab xy* **by** *auto*
ultimately have ∃ *w*∈*Epigraph UNIV f*. *dist w* (*x*, *z*) < *e* **by** *auto*
} **hence** (*x*,*z*):*closure* (*Epigraph UNIV f*) **using** *closure-approachable* **by** *auto*
}
hence ∀ *x y z*. (*x*,*y*) ∈ *closure* (*Epigraph UNIV f*) ∧ *y*≤*z* ⟶ (*x*,*z*) ∈ *closure* (*Epigraph UNIV f*) **by** *auto*
from *this* **show** *?thesis* **using** *mono-epigraph*[*of closure* (*Epigraph UNIV f*)] **by** *auto*
qed

lemma *epigraph-invertible*:
assumes *Epigraph UNIV f* = *Epigraph UNIV g*
shows *f*=*g*
proof−
{ **fix** *x*
{ **fix** *C* **have** *f x* ≤ *ereal C* ⟷ (*x*,*C*) : *Epigraph UNIV f* **unfolding** *Epigraph-def* **by** *auto*
also have ... ⟷ (*x*,*C*) : *Epigraph UNIV g* **using** *assms* **by** *auto*
also have ... ⟷ *g x* ≤ *ereal C* **unfolding** *Epigraph-def* **by** *auto*
finally have *f x* ≤ *ereal C* ⟷ *g x* ≤ *ereal C* **by** *auto*
} **hence** *g x* = *f x* **using** *ereal-le-real*[*of g x f x*] *ereal-le-real*[*of f x g x*] **by** *auto*
} **from** *this* **show** *?thesis* **by** (*simp add*: *ext*)
qed

lemma *lsc-hull-ex-unique*:
fixes *f* :: *'a*::*metric-space* ⇒ *ereal*
shows ∃!*g*. *Epigraph UNIV g* = *closure* (*Epigraph UNIV f*)
using *lsc-hull-exists epigraph-invertible* **by** *metis*

lemma *epigraph-lsc-hull*:
fixes *f* :: *'a*::*metric-space* ⇒ *ereal*
shows *Epigraph UNIV* (*lsc-hull f*) = *closure*(*Epigraph UNIV f*)
proof−
have ∃ *g*. *Epigraph UNIV g* = *closure* (*Epigraph UNIV f*) **using** *lsc-hull-exists* **by** *auto*
thus *?thesis* **unfolding** *lsc-hull-def*
using *some-eq-ex*[*of* (λ*g*. *Epigraph UNIV g* = *closure*(*Epigraph UNIV f*))] **by** *auto*
qed

```
lemma lsc-hull-expl:
  (g = lsc-hull f) ⟷ (Epigraph UNIV g = closure(Epigraph UNIV f))
proof−
{ assume Epigraph UNIV g = closure(Epigraph UNIV f)
  hence lsc-hull f = g unfolding lsc-hull-def apply (subst some1-equality[of - g])
    using lsc-hull-ex-unique by metis+
}
from this show ?thesis using epigraph-lsc-hull by auto
qed


lemma lsc-lsc-hull: lsc (lsc-hull f)
   using epigraph-lsc-hull[of f] lsc-iff[of lsc-hull f] by auto


lemma epigraph-subset-iff:
  fixes f g :: 'a::metric-space ⇒ ereal
  shows Epigraph UNIV f ≤ Epigraph UNIV g ⟷ (∀ x. g x ≤ f x)
proof−
{ assume epi: Epigraph UNIV f ≤ Epigraph UNIV g
  { fix x
    { fix z assume f x ≤ ereal z
      hence (x,z)∈Epigraph UNIV f unfolding Epigraph-def by auto
      hence (x,z)∈Epigraph UNIV g using epi by auto
      hence g x ≤ ereal z unfolding Epigraph-def by auto
    } hence g x ≤ f x apply (subst ereal-le-real) by auto
  }
}
moreover
{ assume le: ∀ x. g x ≤ f x
  { fix x y assume (x,y):Epigraph UNIV f
    hence f x ≤ ereal y unfolding Epigraph-def by auto
    moreover have g x ≤ f x using le by auto
    ultimately have g x ≤ ereal y by auto
    hence (x,y):Epigraph UNIV g unfolding Epigraph-def by auto
  }
}
ultimately show ?thesis by auto
qed


lemma lsc-hull-le: (lsc-hull f) x ≤ f x
  using epigraph-lsc-hull[of f] closure-subset epigraph-subset-iff[of f lsc-hull f] by
auto


lemma lsc-hull-greatest:
fixes f g :: 'a::metric-space ⇒ ereal
```

assumes *lsc g* $\forall x.\ g\ x \leq f\ x$
shows $\forall x.\ g\ x \leq$ (*lsc-hull f*) x
proof−
have *closure*(*Epigraph UNIV f*) $\leq$ *Epigraph UNIV g*
  using *lsc-iff epigraph-subset-iff assms* **by** (*metis closure-minimal*)
from *this* **show** *?thesis* **using** *epigraph-subset-iff lsc-hull-expl* **by** *metis*
qed

lemma *lsc-hull-iff-greatest*:
fixes *f g* :: *'a*::*metric-space* $\Rightarrow$ *ereal*
shows (g = *lsc-hull f*) $\longleftrightarrow$
  lsc g $\wedge$ ($\forall x.\ g\ x \leq f\ x$) $\wedge$ ($\forall h.$ *lsc h* $\wedge$ ($\forall x.\ h\ x \leq f\ x$) $\longrightarrow$ ($\forall x.\ h\ x \leq g\ x$))
(**is** *?lhs* $\longleftrightarrow$ *?rhs*)
proof−
{ **assume** *?lhs* **hence** *?rhs* **using** *lsc-lsc-hull lsc-hull-le lsc-hull-greatest* **by** *metis*
}
moreover
{ **assume** *?rhs*
  { **fix** x **have** (*lsc-hull f*) $x \leq g\ x$ **using** ‹*?rhs*› *lsc-lsc-hull lsc-hull-le* **by** *metis*
    moreover have (*lsc-hull f*) $x \geq g\ x$ **using** ‹*?rhs*› *lsc-hull-greatest* **by** *metis*
    ultimately have (*lsc-hull f*) $x = g\ x$ **by** *auto*
  } **hence** *?lhs* **by** (*simp add*: *ext*)
} **ultimately show** *?thesis* **by** *blast*
qed

lemma *lsc-hull-mono*:
fixes *f g* :: *'a*::*metric-space* $\Rightarrow$ *ereal*
assumes $\forall x.\ g\ x \leq f\ x$
shows $\forall x.$ (*lsc-hull g*) $x \leq$ (*lsc-hull f*) x
proof−
{ **fix** x **have** (*lsc-hull g*) $x \leq g\ x$ **using** *lsc-hull-le*[*of g x*] **by** *auto*
  also have ... $\leq f\ x$ **using** *assms* **by** *auto*
  finally have (*lsc-hull g*) $x \leq f\ x$ **by** *auto*
} **note** $*$ = *this*
show *?thesis* **apply** (*subst lsc-hull-greatest*) **using** *lsc-lsc-hull*[*of g*] $*$ **by** *auto*
qed

lemma *lsc-hull-lsc*:
  lsc f $\longleftrightarrow$ (f = *lsc-hull f*)
using *lsc-hull-iff-greatest*[*of f f*] **by** *auto*

lemma *lsc-hull-liminf-at*:
  fixes f :: *'a*::*metric-space* $\Rightarrow$ *ereal*
  shows $\forall x.$ (*lsc-hull f*) x = *min* ($f\ x$) (*Liminf* (*at x*) f)
proof−

{ **fix** x z **assume** (x,z):Epigraph UNIV (λx. min (f x) (Liminf (at x) f))
hence xz-def: ereal z ≥ (SUP e∈{0<..}. INF y∈ball x e. f y)
unfolding Epigraph-def min-Liminf-at **by** auto
{ **fix** e::real **assume** e>0
hence e/sqrt 2>0 **using** ‹e>0› **by** simp
from this **obtain** e1 **where** e1-def: e1<e/sqrt 2 ∧ e1>0 **using** dense **by** auto
hence (SUP e∈{0<..}. INF y∈ball x e. f y) ≥ (INF y∈ball x e1. f y)
by (auto intro: SUP-upper)
hence ereal z ≥ (INF y∈ball x e1. f y) **using** xz-def **by** auto
hence ∗: ∀ y>ereal z. ∃ t. t ∈ ball x e1 ∧ f t ≤ y
by (simp add: Bex-def Inf-le-iff-less)
obtain t **where** t-def: t : ball x e1 ∧ f t ≤ ereal(z+e1)
using e1-def ∗[rule-format, of ereal(z+e1)] **by** auto
hence epi: (t,z+e1):Epigraph UNIV f **unfolding** Epigraph-def **by** auto
have dist x t < e1 **using** t-def **unfolding** ball-def dist-norm **by** auto
hence sqrt (e1 ^ 2 + dist x t ^ 2) < e
using e1-def **apply** (subst real-sqrt-sum-squares-less) **by** auto
moreover have dist (x,z) (t,z+e1) = sqrt (e1 ^ 2 + dist x t ^ 2)
unfolding dist-prod-def dist-norm **by** (simp add: algebra-simps)
ultimately have dist (x,z) (t,z+e1) < e **by** auto
hence ∃ y∈Epigraph UNIV f. dist y (x, z) < e
apply (rule-tac x=(t,z+e1) **in** bexI) **apply** (simp add: dist-commute) **using** epi **by** auto
} **hence** (x,z) : closure (Epigraph UNIV f)
using closure-approachable[of (x,z) Epigraph UNIV f] **by** auto
}
moreover
{ **fix** x z **assume** xz-def: (x,z):closure (Epigraph UNIV f)
{ **fix** e::real **assume** e>0
from this **obtain** y **where** y-def: y:(Epigraph UNIV f) ∧ dist y (x,z) < e
using closure-approachable[of (x,z) Epigraph UNIV f] xz-def **by** metis
have dist (fst y) x ≤ sqrt ((dist (fst y) x) ^ 2 + (dist (snd y) z) ^ 2)
by (auto intro: real-sqrt-sum-squares-ge1)
also have ...<e **using** y-def **unfolding** dist-prod-def **by** (simp add: algebra-simps)
finally have dist (fst y) x < e **by** auto
hence h1: fst y:ball x e **unfolding** ball-def **by** (simp add: dist-commute)
have dist (snd y) z ≤ sqrt ((dist (fst y) x) ^ 2 + (dist (snd y) z) ^ 2)
by (auto intro: real-sqrt-sum-squares-ge2)
also have ...<e **using** y-def **unfolding** dist-prod-def **by** (simp add: algebra-simps)
finally have h2: dist (snd y) z < e **by** auto
have (INF y∈ball x e. f y) ≤ f(fst y) **using** h1 **by** (simp add: INF-lower)
also have ...≤ereal(snd y) **using** y-def **unfolding** Epigraph-def **by** auto
also have ... < ereal(z+e) **using** h2 **unfolding** dist-norm **by** auto
finally have (INF y∈ball x e. f y) < ereal(z+e) **by** auto
} **hence** ∗: ∀ e>0. (INF y∈ball x e. f y) < ereal(z+e) **by** auto

{ **fix** e **assume** (e::real)>0

{ **fix** *d* **assume** (*d*::*real*)>*0*
{ **assume** *d*<*e*
have (*INF y*∈*ball x d. f y*) < *ereal*(*z*+*d*) **using** ∗ ‹*d*>*0*› **by** *auto*
also have . . . < *ereal*(*z*+*e*) **using** ‹*d*<*e*› **by** *auto*
finally have (*INF y*∈*ball x d. f y*) < *ereal*(*z*+*e*) **by** *auto*
}
moreover
{ **assume** ~(*d*<*e*)
hence *ball x e* ≤ *ball x d* **by** *auto*
hence (*INF y*∈*ball x d. f y*) ≤ (*INF y*∈*ball x e. f y*) **apply** (*subst INF-mono*) **by** *auto*
also have . . . < *ereal*(*z*+*e*) **using** ∗ ‹*e*>*0*› **by** *auto*
finally have (*INF y*∈*ball x d. f y*) < *ereal*(*z*+*e*) **by** *auto*
} **ultimately have** (*INF y*∈*ball x d. f y*) < *ereal*(*z*+*e*) **by** *blast*
hence (*INF y*∈*ball x d. f y*) ≤ *ereal*(*z*+*e*) **by** *auto*
} **hence** *min* (*f x*) (*Liminf* (*at x*) *f*) ≤ *ereal* (*z*+*e*) **unfolding** *min-Liminf-at* **by** (*auto intro*: *SUP-least*)
} **hence** *min* (*f x*) (*Liminf* (*at x*) *f*) ≤ *ereal z* **using** *ereal-le-epsilon2* **by** *auto*
hence (*x*,*z*):*Epigraph UNIV* (λ*x*. *min* (*f x*) (*Liminf* (*at x*) *f*)) **unfolding** *Epigraph-def* **by** *auto*
}
ultimately have *Epigraph UNIV* (λ*x*. *min* (*f x*) (*Liminf* (*at x*) *f*))= *closure* (*Epigraph UNIV f*) **by** *auto*
hence (λ*x*. *min* (*f x*) (*Liminf* (*at x*) *f*)) = *lsc-hull f*
using *epigraph-invertible epigraph-lsc-hull*[*of f*] **by** *blast*
from *this* **show** *?thesis* **by** *metis*
qed

lemma *lsc-hull-same-inf*:
fixes *f* :: ′*a*::*metric-space* ⇒ *ereal*
shows (*INF x*. *lsc-hull f x*) = (*INF x*. *f x*)
proof−
{ **fix** *x*
have (*INF x*. *f x*) ≤ (*INF y*∈*ball x 1*. *f y*) **apply** (*subst INF-mono*) **by** *auto*
also have . . . ≤ *min* (*f x*) (*Liminf* (*at x*) *f*) **unfolding** *min-Liminf-at* **by** (*auto intro*: *SUP-upper*)
also have . . . =(*lsc-hull f*) *x* **using** *lsc-hull-liminf-at*[*of f*] **by** *auto*
finally have (*INF x*. *f x*) ≤ (*lsc-hull f*) *x* **by** *auto*
} **hence** (*INF x*. *f x*) ≤ (*INF x*. *lsc-hull f x*) **apply** (*subst INF-greatest*) **by** *auto*
moreover have (*INF x*. *lsc-hull f x*) ≤ (*INF x*. *f x*)
apply (*subst INF-mono*) **using** *lsc-hull-le* **by** *auto*
ultimately show *?thesis* **by** *auto*
qed

1.4 Convex Functions

definition
convex-on :: ′*a*::*real-vector set* ⇒ (′*a* ⇒ *ereal*) ⇒ *bool* **where**

```
  convex-on s f ⟷
  (∀x∈s. ∀y∈s. ∀u≥0. ∀v≥0. u + v = 1
  ⟶ f (u *R x + v *R y) ≤ ereal u * f x + ereal v * f y)
```

```
lemma convex-on-ereal-mem:
  assumes convex-on s f
  assumes x:s y:s
  assumes u≥0 v≥0 u+v=1
  shows f (u *R x + v *R y) ≤ ereal u * f x + ereal v * f y
using assms unfolding convex-on-def by auto
```

```
lemma convex-on-ereal-subset: convex-on t f ⟹ s ≤ t ⟹ convex-on s f
  unfolding convex-on-def by auto
```

```
lemma convex-on-ereal-univ: convex-on UNIV f ⟷ (∀S. convex-on S f)
  using convex-on-ereal-subset by auto
```

```
lemma ereal-pos-sum-distrib-left:
  fixes f :: 'a ⇒ ereal
  assumes r≥0 r ≠ ∞
  shows r * sum f A = sum (λn. r * f n) A
proof (cases finite A)
  case True
  thus ?thesis
  proof induct
    case empty thus ?case by simp
  next
    case (insert x A) thus ?case using assms by (simp add: ereal-pos-distrib)
  qed
next
  case False thus ?thesis by simp
qed
```

```
lemma convex-ereal-add:
  fixes f g :: 'a::real-vector ⇒ ereal
  assumes convex-on s f convex-on s g
  shows convex-on s (λx. f x + g x)
proof−
  { fix x y assume x:s y:s moreover
    fix u v ::real assume uv: 0 ≤ u 0 ≤ v u + v = 1
    ultimately have f (u *R x + v *R y) + g (u *R x + v *R y)
      ≤ (ereal u * f x + ereal v * f y) + (ereal u * g x + ereal v * g y)
      using assms unfolding convex-on-def by (auto simp add: add-mono)
    also have ... =(ereal u * f x + ereal u * g x) + (ereal v * f y + ereal v * g y)
      by (simp add: algebra-simps)
```

also have ... = *ereal u* $*$ (*f x* + *g x*) + *ereal v* $*$ (*f y* + *g y*)
using *uv* **by** (*simp add*: *ereal-pos-distrib*)
finally have *f* ($u *_R x + v *_R y$) + *g* ($u *_R x + v *_R y$)
$\leq$ *ereal u* $*$ (*f x* + *g x*) + *ereal v* $*$ (*f y* + *g y*) **by** *auto* **}**
thus *?thesis* **unfolding** *convex-on-def* **by** *auto*
qed

lemma *convex-ereal-cmul*:
assumes *0* $\leq$ (*c*::*ereal*) *convex-on s f*
shows *convex-on s* (λx. *c* $*$ *f x*)
proof−
{ fix *x y* **assume** *x*:*s* *y*:*s* **moreover**
fix *u v* ::*real* **assume** *uv*: *0* $\leq$ *u* *0* $\leq$ *v* *u* + *v* = *1*
ultimately have *f* ($u *_R x + v *_R y$) $\leq$ (*ereal u* $*$ *f x* + *ereal v* $*$ *f y*)
using *assms* **unfolding** *convex-on-def* **by** *auto*
hence *c* $*$ *f* ($u *_R x + v *_R y$) $\leq$ *c* $*$ (*ereal u* $*$ *f x* + *ereal v* $*$ *f y*)
using *assms* **by** (*intro ereal-mult-left-mono*) *auto*
also have ... $\leq$ *c* $*$ (*ereal u* $*$ *f x*) + *c* $*$ (*ereal v* $*$ *f y*)
using *assms* **by** (*simp add*: *ereal-le-distrib*)
also have ... = *ereal u* $*$(*c*$*$ *f x*) + *ereal v* $*$(*c*$*$ *f y*) **by** (*simp add*: *algebra-simps*)
finally have *c* $*$ *f* ($u *_R x + v *_R y$)
$\leq$ *ereal u* $*$ (*c* $*$ *f x*) + *ereal v* $*$ (*c* $*$ *f y*) **by** *auto* **}**
thus *?thesis* **unfolding** *convex-on-def* **by** *auto*
qed

lemma *convex-ereal-max*:
fixes *f g* :: *'a*::*real-vector* $\Rightarrow$ *ereal*
assumes *convex-on s f convex-on s g*
shows *convex-on s* (λx. *max* (*f x*) (*g x*))
proof−
{ fix *x y* **assume** *x*:*s* *y*:*s* **moreover**
fix *u v* ::*real* **assume** *uv*: *0* $\leq$ *u* *0* $\leq$ *v* *u* + *v* = *1*
ultimately have *max* (*f* ($u *_R x + v *_R y$)) (*g* ($u *_R x + v *_R y$))
$\leq$ *max* (*ereal u* $*$ *f x* + *ereal v* $*$ *f y*) (*ereal u* $*$ *g x* + *ereal v* $*$ *g y*)
apply (*subst max.mono*) **using** *assms* **unfolding** *convex-on-def* **by** *auto*
also have ... $\leq$ *ereal u* $*$ *max* (*f x*) (*g x*) + *ereal v* $*$ *max* (*f y*) (*g y*)
apply (*subst max.boundedI*)
apply (*subst add-mono*) **prefer** *4* **apply** (*subst add-mono*)
by (*subst ereal-mult-left-mono*, *auto simp add*: *uv*)+
finally have *max* (*f* ($u *_R x + v *_R y$)) (*g* ($u *_R x + v *_R y$))
$\leq$ *ereal u* $*$ *max* (*f x*) (*g x*) + *ereal v* $*$ *max* (*f y*) (*g y*) **by** *auto* **}**
thus *?thesis* **unfolding** *convex-on-def* **by** *auto*
qed

lemma *convex-on-ereal-alt*:
fixes *C* :: *'a*::*real-vector set*

assumes *convex C*
shows *convex-on C f* =
(∀ *x* ∈ *C*. ∀ *y* ∈ *C*. ∀ *m* :: *real*. *m* ≥ *0* ∧ *m* ≤ *1*
⟶ *f* (*m* $*_R$ *x* + (*1* − *m*) $*_R$ *y*) ≤ (*ereal m*) ∗ *f x* + (*1* − (*ereal m*)) ∗ *f y*)
proof *safe*
fix *x y* **fix** *m* :: *real*
have[*simp*]: *ereal* (*1* − *m*) = (*1* − *ereal m*)
using *ereal-minus*(*1*)[*of 1 m*] **by** (*auto simp*: *one-ereal-def*)
assume *asms*: *convex-on C f x* : *C y* : *C 0* ≤ *m m* ≤ *1*
from *this*[*unfolded convex-on-def*, *rule-format*]
have ∀ *u v*. ((*0* ≤ *u* ∧ *0* ≤ *v* ∧ *u* + *v* = *1*) ⟶
f (*u* $*_R$ *x* + *v* $*_R$ *y*) ≤ (*ereal u*) ∗ *f x* + (*ereal v*) ∗ *f y*) **by** *auto*
from *this*[*rule-format*, *of m 1* − *m*, *simplified*] *asms*
show *f* (*m* $*_R$ *x* + (*1* − *m*) $*_R$ *y*)
≤ (*ereal m*) ∗ *f x* + (*1* − *ereal m*) ∗ *f y* **by** *auto*
next
assume *asm*: ∀ *x*∈*C*. ∀ *y*∈*C*. ∀ *m*. *0* ≤ *m* ∧ *m* ≤ *1*
⟶ *f* (*m* $*_R$ *x* + (*1* − *m*) $*_R$ *y*) ≤ (*ereal m*) ∗ *f x* + (*1* − *ereal m*) ∗ *f y*
{ fix *x y* **fix** *u v* :: *real*
assume *lasm*: *x* ∈ *C y* ∈ *C u* ≥ *0 v* ≥ *0 u* + *v* = *1*
hence[*simp*]: *1*−*u*=*v 1* − *ereal u* = *ereal v*
using *ereal-minus*(*1*)[*of 1 m*] **by** (*auto simp*: *one-ereal-def*)
from *asm*[*rule-format*, *of x y u*]
have *f* (*u* $*_R$ *x* + *v* $*_R$ *y*) ≤ (*ereal u*) ∗ *f x* + (*ereal v*) ∗ *f y*
using *lasm* **by** *auto* **}**
thus *convex-on C f* **unfolding** *convex-on-def* **by** *auto*
qed

lemma *convex-on-ereal-alt-mem*:
fixes *C* :: ′*a*::*real-vector set*
assumes *convex C*
assumes *convex-on C f*
assumes *x* : *C y* : *C*
assumes (*m*::*real*) ≥ *0 m* ≤ *1*
shows *f* (*m* $*_R$ *x* + (*1* − *m*) $*_R$ *y*) ≤ (*ereal m*) ∗ *f x* + (*1* − (*ereal m*)) ∗ *f y*
by (*metis assms convex-on-ereal-alt*)

lemma *ereal-add-right-mono*: (*a*::*ereal*) ≤ *b* ⟹ *a* + *c* ≤ *b* + *c*
by (*metis add-mono order-refl*)

lemma *convex-on-ereal-sum-aux*:
assumes *1*−*a*>*0*
shows (*1* − *ereal a*) ∗ (*ereal* (*c* / (*1* − *a*)) ∗ *b*) = (*ereal c*) ∗ *b*
by (*metis mult.assoc mult.commute eq-divide-eq ereal-minus*(*1*) *assms*
one-ereal-def less-le times-ereal.simps(*1*))

```
lemma convex-on-ereal-sum:
  fixes a :: 'a ⇒ real
  fixes y :: 'a ⇒ 'b::real-vector
  fixes f :: 'b ⇒ ereal
  assumes finite s s ≠ {}
  assumes convex-on C f
  assumes convex C
  assumes (SUM i : s. a i) = 1
  assumes ∀ i. i ∈ s ⟶ a i ≥ 0
  assumes ∀ i. i ∈ s ⟶ y i ∈ C
  shows f (SUM i : s. a i *R y i) ≤ (SUM i : s. ereal (a i) * f (y i))
using assms(1,2,5−)
proof (induct s arbitrary:a rule:finite-ne-induct)
  case (singleton i)
  hence ai: a i = 1 by auto
  thus ?case by (auto simp: one-ereal-def[symmetric])
next
  case (insert i s)
  from ‹convex-on C f›
  have conv: ∀ x y m. ((x ∈ C ∧ y ∈ C ∧ 0 ≤ m ∧ m ≤ 1)
      ⟶ f (m *R x + (1 − m) *R y) ≤ (ereal m) * f x + (1 − ereal m) * f y)
    using convex-on-ereal-alt[of C f] ‹convex C› by auto
  { assume a i = 1
    hence (SUM j : s. a j) = 0
      using insert by auto
    hence ∀ j. (j ∈ s ⟶ a j = 0)
      using insert by (simp add: sum-nonneg-eq-0-iff)
    hence ?case using insert.hyps(1−3) ‹a i = 1›
      by (simp add: zero-ereal-def[symmetric] one-ereal-def[symmetric]) }
  moreover
  { assume asm: a i ≠ 1
    from insert have yai: y i : C a i ≥ 0 by auto
    have fis: finite (insert i s) using insert by auto
    hence ai1: a i ≤ 1 using sum-nonneg-leq-bound[of insert i s a] insert by simp
    hence a i < 1 using asm by auto
    hence i0: 1 − a i > 0 by auto
    hence i1: 1 − ereal (a i) > 0 using ereal-minus(1)[of 1 a i]
      by (simp add: zero-ereal-def[symmetric] one-ereal-def[symmetric])
    have i2: 1 − ereal (a i) ≠ ∞ using ereal-minus(1)[of 1]
      by (simp add: zero-ereal-def[symmetric] one-ereal-def[symmetric])
    let ?a j = a j / (1 − a i)
    have a-nonneg: ⋀j. j ∈ s ⟹ 0 ≤ a j / (1 − a i)
      using i0 insert
      by (metis insert-iff divide-nonneg-pos)
    have (SUM j : insert i s. a j) = 1 using insert by auto
    hence (SUM j : s. a j) = 1 − a i using sum.insert insert by fastforce
    hence (SUM j : s. a j) / (1 − a i) = 1 using i0 by auto
    hence a1: (SUM j : s. ?a j) = 1 unfolding sum-divide-distrib by simp
    have asum: (SUM j : s. ?a j *R y j) : C
```

using *insert convex-sum*[*OF* ‹*finite s*›
‹*convex C*› *a1 a-nonneg*] **by** *auto*
have *asum-le*: *f* (*SUM j* : *s*. *?a j* $*_R$ *y j*) ≤ (*SUM j* : *s*. *ereal* (*?a j*) ∗ *f* (*y j*))
using *a-nonneg a1 insert* **by** *blast*
have *f* (*SUM j* : *insert i s*. *a j* $*_R$ *y j*) = *f* ((*SUM j* : *s*. *a j* $*_R$ *y j*) + *a i* $*_R$ *y i*)
using *sum.insert*[*of s i* λ*j*. *a j* $*_R$ *y j*, *OF* ‹*finite s*› ‹*i* ∉ *s*›]
by (*auto simp only*:*add.commute*)
also have ... = *f* (((*1* − *a i*) ∗ *inverse* (*1* − *a i*)) $*_R$ (*SUM j* : *s*. *a j* $*_R$ *y j*) + *a i* $*_R$ *y i*)
using *i0* **by** *auto*
also have ... = *f* ((*1* − *a i*) $*_R$ (*SUM j* : *s*. (*a j* ∗ *inverse* (*1* − *a i*)) $*_R$ *y j*) + *a i* $*_R$ *y i*)
using *scaleR-right.sum*[*of inverse* (*1* − *a i*) λ*j*. *a j* $*_R$ *y j s*, *symmetric*] **by** (*auto simp*:*algebra-simps*)
also have ... = *f* ((*1* − *a i*) $*_R$ (*SUM j* : *s*. *?a j* $*_R$ *y j*) + *a i* $*_R$ *y i*)
by (*auto simp*: *divide-inverse*)
also have ... ≤ (*1* − *ereal* (*a i*)) ∗ *f* ((*SUM j* : *s*. *?a j* $*_R$ *y j*)) + (*ereal* (*a i*)) ∗ *f* (*y i*)
using *conv*[*rule-format*, *of y i* (*SUM j* : *s*. *?a j* $*_R$ *y j*) *a i*]
using *yai*(*1*) *asum yai*(*2*) *ai1* **by** (*auto simp add*:*add.commute*)
also have ... ≤ (*1* − *ereal* (*a i*)) ∗ (*SUM j* : *s*. *ereal* (*?a j*) ∗ *f* (*y j*)) + (*ereal* (*a i*)) ∗ *f* (*y i*)
using *ereal-add-right-mono*[*OF ereal-mult-left-mono*[*of* - - *1* − *ereal* (*a i*),
OF asum-le less-imp-le[*OF i1*]], *of* (*ereal* (*a i*)) ∗ *f* (*y i*)] **by** *simp*
also have ... = (*SUM j* : *s*. (*1* − *ereal* (*a i*)) ∗ (*ereal* (*?a j*) ∗ *f* (*y j*))) + (*ereal* (*a i*)) ∗ *f* (*y i*)
unfolding *ereal-pos-sum-distrib-left*[*of 1* − *ereal* (*a i*) λ*j*. (*ereal* (*?a j*)) ∗ *f* (*y j*), *OF less-imp-le*[*OF i1*] *i2*] **by** *auto*
also have ... = (*SUM j* : *s*. (*ereal* (*a j*)) ∗ *f* (*y j*)) + (*ereal* (*a i*)) ∗ *f* (*y i*) **using** *i0 convex-on-ereal-sum-aux* **by** *auto*
also have ... = (*ereal* (*a i*)) ∗ *f* (*y i*) + (*SUM j* : *s*. (*ereal* (*a j*)) ∗ *f* (*y j*)) **by** (*simp add*: *add.commute*)
also have ... = (*SUM j* : *insert i s*. (*ereal* (*a j*)) ∗ *f* (*y j*)) **using** *insert* **by** *auto*
finally have *f* (*SUM j* : *insert i s*. *a j* $*_R$ *y j*) ≤ (*SUM j* : *insert i s*. (*ereal* (*a j*)) ∗ *f* (*y j*)) **by** *simp* **}**
ultimately show *?case* **by** *auto*
qed

lemma *sum-2*: *sum u* {*1*::*nat*..*2*} = (*u 1*)+(*u 2*)
proof−
have {*1*::*nat*..*2*} = {*1*::*nat*,*2*} **by** *auto*
thus *?thesis* **by** *auto*
qed

lemma *convex-on-ereal-iff*:

assumes *convex s*
shows *convex-on s f* $\longleftrightarrow$ ($\forall$ *k u x*. ($\forall$ *i*$\in${*1..k::nat*}. *0* $\leq$ *u i* $\wedge$ *x i* : *s*) $\wedge$ *sum u* {*1..k*} = *1* $\longrightarrow$
f (*sum* (λ*i*. *u i* $*_R$ *x i*) {*1..k*}) $\leq$ *sum* (λ*i*. (*ereal* (*u i*)) $*$ *f*(*x i*)) {*1..k*})
(**is** *?rhs* $\longleftrightarrow$ *?lhs*)
proof−
{ **assume** *?rhs*
{ **fix** *k u x*
have *zero*: ~(*sum u* {*1..0::nat*} = (*1::real*)) **by** *auto*
assume ($\forall$ *i*$\in${*1..k::nat*}. (*0::real*) $\leq$ *u i* $\wedge$ *x i* $\in$ *s*)
moreover assume $*$: *sum u* {*1..k*} = *1*
moreover from $*$ **have** *k* $\neq$ *0* **using** *zero* **by** *metis*
ultimately have *f* (*sum* (λ*i*. *u i* $*_R$ *x i*) {*1..k*})
$\leq$ *sum* (λ*i*. (*ereal* (*u i*)) $*$ *f*(*x i*)) {*1..k*}
using *convex-on-ereal-sum*[*of* {*1..k*} *s f u x*] **using** *assms* ‹*?rhs*› **by** *auto*
} **hence** *?lhs* **by** *auto*
}
moreover
{ **assume** *?lhs*
{ **fix** *x y u v*
assume *xys*: *x*:*s y*:*s*
assume *uv1*: *u*$\geq$*0 v*$\geq$*0 u* + *v* = (*1::real*)
define *xy* **where** *xy* = (λ*i::nat*. *if i=1 then x else y*)
define *uv* **where** *uv* = (λ*i::nat*. *if i=1 then u else v*)
have $\forall$ *i*$\in${*1..2::nat*}. (*0* $\leq$ *uv i*) $\wedge$ (*xy i* : *s*) **unfolding** *xy-def uv-def* **using** *xys uv1* **by** *auto*
moreover have *sum uv* {*1..2*} = *1* **using** *sum-2*[*of uv*] **unfolding** *uv-def* **using** *uv1* **by** *auto*
moreover have (*SUM i* = *1..2*. *uv i* $*_R$ *xy i*) = *u* $*_R$ *x* + *v* $*_R$ *y*
using *sum-2*[*of* (λ*i*. *uv i* $*_R$ *xy i*)] **unfolding** *xy-def uv-def* **using** *xys uv1* **by** *auto*
moreover have (*SUM i* = *1..2*. *ereal* (*uv i*) $*$ *f* (*xy i*)) = *ereal u* $*$ *f x* + *ereal v* $*$ *f y*
using *sum-2*[*of* (λ*i*. *ereal* (*uv i*) $*$ *f* (*xy i*))] **unfolding** *xy-def uv-def* **using** *xys uv1* **by** *auto*
ultimately have *f* (*u* $*_R$ *x* + *v* $*_R$ *y*) $\leq$ *ereal u* $*$ *f x* + *ereal v* $*$ *f y*
using ‹*?lhs*›[*rule-format*, *of 2 uv xy*] **by** *auto*
} **hence** *?rhs* **unfolding** *convex-on-def* **by** *auto*
} **ultimately show** *?thesis* **by** *blast*
qed

lemma *convex-Epigraph*:
assumes *convex S*
shows *convex*(*Epigraph S f*) $\longleftrightarrow$ *convex-on S f*
proof−
{ **assume** *rhs*: *convex*(*Epigraph S f*)
{ **fix** *x y* **assume** *xy*: *x*:*S y*:*S*
fix *u v* ::*real* **assume** *uv*: *0* $\leq$ *u 0* $\leq$ *v u* + *v* = *1*

have $f\ (u *_R x + v *_R y) \leq$ *ereal* $u * f\ x$ + *ereal* $v * f\ y$
proof−
{ assume $u=0 \mid v=0$ **hence** *?thesis* **using** *uv* **by** (*auto simp*: *zero-ereal-def*[*symmetric*] *one-ereal-def*[*symmetric*]) **}**
moreover
{ assume $f\ x = \infty \mid f\ y = \infty$ **hence** *?thesis* **using** *uv* **by** (*auto simp*: *zero-ereal-def*[*symmetric*] *one-ereal-def*[*symmetric*]) **}**
moreover
{ assume *a*: $f\ x = -\infty \wedge (f\ y \neq \infty) \wedge \sim(u=0)$
from *this* **obtain** *z* **where** $f\ y \leq$ *ereal z* **apply** (*cases f y*) **by** *auto*
hence *yz*: (y,z) : *Epigraph S f* **unfolding** *Epigraph-def* **using** *xy* **by** *auto*
{ fix *C*::*real*
have $(x, (1/u)*(C - v * z))$: *Epigraph S f* **unfolding** *Epigraph-def* **using** *a xy* **by** *auto*
hence $(u *_R x + v *_R y, C)$: *Epigraph S f*
using *rhs convexD*[*of Epigraph S f* $(x, (1/u)*(C - v * z))$ (y,z) *u v*] *uv yz a* **by** *auto*
hence $(f\ (u *_R x + v *_R y) \leq$ *ereal* $C)$ **unfolding** *Epigraph-def* **by** *auto*
} hence $f\ (u *_R x + v *_R y) = -\infty$ **using** *ereal-bot* **by** *auto*
hence *?thesis* **by** *auto* **}**
moreover
{ assume *a*: $(f\ x \neq \infty) \wedge f\ y = -\infty \wedge \sim(v=0)$
from *this* **obtain** *z* **where** $f\ x \leq$ *ereal z* **apply** (*cases f x*) **by** *auto*
hence *xz*: (x,z) : *Epigraph S f* **unfolding** *Epigraph-def* **using** *xy* **by** *auto*
{ fix *C*::*real*
have $(y, (1/v)*(C - u * z))$: *Epigraph S f* **unfolding** *Epigraph-def* **using** *a xy* **by** *auto*
hence $(u *_R x + v *_R y, C)$: *Epigraph S f*
using *rhs convexD*[*of Epigraph S f* (x,z) $(y, (1/v)*(C - u * z))$ *u v*] *uv xz a* **by** *auto*
hence $(f\ (u *_R x + v *_R y) \leq$ *ereal* $C)$ **unfolding** *Epigraph-def* **by** *auto*
} hence $f\ (u *_R x + v *_R y) = -\infty$ **using** *ereal-bot* **by** *auto*
hence *?thesis* **by** *auto* **}**
moreover
{ assume *a*: $f\ x \neq \infty \wedge f\ x \neq -\infty \wedge f\ y \neq \infty \wedge f\ y \neq -\infty$
from *this* **obtain** *fx fy* **where** *fxy*: $f\ x$ = *ereal fx* $\wedge$ $f\ y$ = *ereal fy*
apply (*cases f x*, *cases f y*) **by** *auto*
hence (x, fx) : *Epigraph S f* $\wedge$ (y, fy) : *Epigraph S f* **unfolding** *Epigraph-def* **using** *xy* **by** *auto*
hence $(u *_R x + v *_R y, u * fx + v * fy)$: *Epigraph S f*
using *rhs convexD*[*of Epigraph S f* (x,fx) (y,fy) *u v*] *uv* **by** *auto*
hence *?thesis* **unfolding** *Epigraph-def* **using** *fxy* **by** *auto*
} ultimately show *?thesis* **by** *blast*
qed
} hence *convex-on S f* **unfolding** *convex-on-def* **by** *auto*
}
moreover
{ assume *lhs*: *convex-on S f*
{ fix *x y fx fy* **assume** *xy*: (x,fx):*Epigraph S f* (y,fy):*Epigraph S f*

fix *u v ::real* **assume** *uv*: $0 \le u$ $0 \le v$ $u + v = 1$
hence *le*: $f\,x \le ereal\,fx \wedge f\,y \le ereal\,fy$ **using** *xy* **unfolding** *Epigraph-def* **by** *auto*
have $x{:}S \wedge y{:}S$ **using** *xy* **unfolding** *Epigraph-def* **by** *auto*
moreover hence *inS*: $u *_R x + v *_R y : S$ **using** *assms uv convexD*[*of S*] **by** *auto*
ultimately have $f(u *_R x + v *_R y) \le (ereal\ u) * f\,x + (ereal\ v) * f\,y$
using *lhs convex-on-ereal-mem*[*of S f x y u v*] *uv* **by** *auto*
also have $\ldots \le (ereal\ u) * (ereal\ fx) + (ereal\ v) * (ereal\ fy)$
apply (*subst add-mono*) **apply** (*subst ereal-mult-left-mono*)
prefer *4* **apply** (*subst ereal-mult-left-mono*) **using** *le uv* **by** *auto*
also have $\ldots = ereal\ (u * fx + v * fy)$ **by** *auto*
finally have $(u *_R x + v *_R y,\ u * fx + v * fy) : Epigraph\ S\ f$
unfolding *Epigraph-def* **using** *inS* **by** *auto*
} **hence** *convex*(*Epigraph S f*) **unfolding** *convex-def* **by** *auto*
}
ultimately show *?thesis* **by** *auto*
qed

lemma *convex-EpigraphI*:
convex-on s f $\Longrightarrow$ *convex s* $\Longrightarrow$ *convex*(*Epigraph s f*)
unfolding *convex-Epigraph* **by** *auto*

definition
concave-on :: *'a::real-vector set* $\Rightarrow$ (*'a* $\Rightarrow$ *ereal*) $\Rightarrow$ *bool* **where**
concave-on S f $\longleftrightarrow$ *convex-on S* $(\lambda x.\ -\ f\ x)$

definition
finite-on :: *'a::real-vector set* $\Rightarrow$ (*'a* $\Rightarrow$ *ereal*) $\Rightarrow$ *bool* **where**
finite-on S f $\longleftrightarrow$ $(\forall x{\in}S.\ (f\ x \ne \infty \wedge f\ x \ne -\infty))$

definition
affine-on :: *'a::real-vector set* $\Rightarrow$ (*'a* $\Rightarrow$ *ereal*) $\Rightarrow$ *bool* **where**
affine-on S f $\longleftrightarrow$ (*convex-on S f* $\wedge$ *concave-on S f* $\wedge$ *finite-on S f*)

definition
domain (*f*::- $\Rightarrow$ *ereal*) $= \{x.\ f\ x < \infty\}$

lemma *domain-Epigraph-aux*:
assumes $x \ne \infty$
shows $\exists\, r.\ x \le ereal\ r$
using *assms* **by** (*cases x*) *auto*

lemma *domain-Epigraph*:
domain f $= \{x.\ \exists\, y.\ (x,y) \in Epigraph\ UNIV\ f\}$

unfolding *domain-def Epigraph-def* **using** *domain-Epigraph-aux* **by** *auto*

lemma *domain-Epigraph-fst*:
domain f = fst ‘ (Epigraph UNIV f)
proof−
{ fix *x* **assume** *x:domain f*
from *this* **obtain** *y* **where** *(x,y):Epigraph UNIV f* **using** *domain-Epigraph[of f]* **by** *auto*
moreover have *x = fst (x,y)* **by** *auto*
ultimately have *x:fst ‘ (Epigraph UNIV f)* **unfolding** *image-def* **by** *blast*
} from *this* **show** *?thesis* **using** *domain-Epigraph[of f]* **by** *auto*
qed

lemma *convex-on-domain*:
convex-on (domain f) f $\longleftrightarrow$ *convex-on UNIV f*
proof−
{ assume *lhs*: *convex-on (domain f) f*
{ fix *x y*
fix *u v ::real* **assume** *uv*: $0 \leq u$ $0 \leq v$ $u + v = 1$
have $f\ (u *_R x + v *_R y) \leq ereal\ u * f\ x + ereal\ v * f\ y$
proof−
{ assume $f\ x = \infty \mid f\ y = \infty$ **hence** *?thesis* **using** *uv* **by** (*auto simp*: *zero-ereal-def[symmetric] one-ereal-def[symmetric]*) **}**
moreover
{ assume ~ $(f\ x = \infty \mid f\ y = \infty)$
hence x : *domain f* $\wedge$ y : *domain f* **unfolding** *domain-def* **by** *auto*
hence *?thesis* **using** *lhs* **unfolding** *convex-on-def* **using** *uv* **by** *auto*
} ultimately show *?thesis* **by** *blast*
qed }
hence *convex-on UNIV f* **unfolding** *convex-on-def* **by** *auto*
} from *this* **show** *?thesis* **using** *convex-on-ereal-subset* **by** *auto*
qed

lemma *convex-on-domain2*:
convex-on (domain f) f $\longleftrightarrow$ ($\forall$ *S. convex-on S f*)
by (*metis convex-on-domain convex-on-ereal-univ*)

lemma *convex-domain*:
fixes *f* :: *'a::euclidean-space* $\Rightarrow$ *ereal*
assumes *convex-on UNIV f*
shows *convex (domain f)*
proof−
have *convex (Epigraph UNIV f)* **using** *assms convex-Epigraph* **by** *auto*
thus *?thesis* **unfolding** *domain-Epigraph-fst*
apply (*subst convex-linear-image*) **using** *linear-fst linear-conv-bounded-linear* **by**

auto
qed

lemma *infinite-convex-domain-iff*:
  fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
  assumes $\forall x.\ (f\ x = \infty \mid f\ x = -\infty)$
  shows *convex-on UNIV f* ⟷ *convex* (*domain f*)
proof−
{ **assume** *dom*: *convex* (*domain f*)
  { **fix** *x y* **assume** *x*:*domain f* *y*:*domain f* **moreover**
    fix *u v* ::*real* **assume** *uv*: $0 \le u$ $0 \le v$ $u + v = 1$
    ultimately have $u *_R x + v *_R y$: *domain f*
      using *dom* **unfolding** *convex-def* **by** *auto*
    hence $f(u *_R x + v *_R y) = -\infty$
      using *assms* **unfolding** *domain-def* **by** *auto*
  } **hence** *convex-on* (*domain f*) *f* **unfolding** *convex-on-def* **by** *auto*
  hence *convex-on UNIV f* **by** (*metis convex-on-domain2*)
} **thus** *?thesis* **by** (*metis convex-domain*)
qed

lemma *convex-PInfty-outside*:
  fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
  assumes *convex-on UNIV f convex S*
  shows *convex-on UNIV* ($\lambda x.$ *if x*:*S then* (*f x*) *else* ∞)
proof−
  define *g* **where** *g x* = (*if x*:*S then* $-\infty$ *else* ∞::*ereal*) **for** *x*
  hence *convex-on UNIV g* **apply** (*subst infinite-convex-domain-iff*)
    using *assms* **unfolding** *domain-def* **by** *auto*
  moreover have ($\lambda x.$ *if x*:*S then* (*f x*) *else* ∞) = ($\lambda x.$ *max* (*f x*) (*g x*))
    apply (*subst fun-eq-iff*) **unfolding** *g-def* **by** *auto*
  ultimately show *?thesis* **using** *convex-ereal-max assms* **by** *auto*
qed

definition
  proper-on :: *'a::real-vector set* ⇒ (*'a* ⇒ *ereal*) ⇒ *bool* **where**
  proper-on S f ⟷ $((\forall x{\in}S.\ f\ x \neq -\infty) \wedge (\exists x{\in}S.\ f\ x \neq \infty))$

definition
  proper :: (*'a::real-vector* ⇒ *ereal*) ⇒ *bool* **where**
  proper f ⟷ *proper-on UNIV f*

lemma *proper-iff*:
  proper f ⟷ $((\forall x.\ f\ x \neq -\infty) \wedge (\exists x.\ f\ x \neq \infty))$
  unfolding *proper-def proper-on-def* **by** *auto*

lemma *improper-iff*:
  $^\sim$(*proper f*) $\longleftrightarrow$ (($\exists$ *x*. *f x* = $-\infty$) | ($\forall$ *x*. *f x* = ∞))
  by (*simp add*: *proper-iff*)

lemma *ereal-MInf-plus*[*simp*]: $-\infty$ + *x* = (*if x* = ∞ *then* ∞ *else* $-\infty$::*ereal*)
  by *simp*

lemma *convex-improper*:
  fixes *f* :: *'a*::*euclidean-space* $\Rightarrow$ *ereal*
  assumes *convex-on UNIV f*
  assumes $^\sim$(*proper f*)
  shows $\forall$ *x*$\in$*rel-interior*(*domain f*). *f x* = $-\infty$
proof−
{ assume *domain f* = {} **hence** *?thesis* **using** *rel-interior-empty* **by** *auto* **}**
moreover
{ assume *nemp*: *domain f* $\neq$ {}
 then obtain *u* **where** *u-def*: *f u* = $-\infty$ **using** *assms improper-iff*[*of f*] **unfolding** *domain-def* **by** *auto*
  hence *udom*: *u*:*domain f* **unfolding** *domain-def* **by** *auto*
  { fix *x* **assume** *x*:*rel-interior*(*domain f*)
    then obtain *m* **where** *m-def*: *m*>*1* $\wedge$ (*1* − *m*) $*_R$ *u* + *m* $*_R$ *x* : *domain f*
      using *convex-rel-interior-iff*[*of domain f x*] *nemp convex-domain*[*of f*] *assms udom* **by** *auto*
    define *v* **where** *v* = *1*/*m*
    hence *v01*: *0*<*v* $\wedge$ *v*<*1* **using** *m-def* **by** *auto*
    define *y* **where** *y* = (*1* − *m*) $*_R$ *u* + *m* $*_R$ *x*
    have *x* = (*1*−*v*) $*_R$ *u*+*v* $*_R$ *y* **unfolding** *v-def y-def* **using** *m-def* **by** (*simp add*: *algebra-simps*)
    hence *f x* $\leq$ (*1*−*ereal v*) $*$ *f u*+*ereal v* $*$ *f y*
      using *convex-on-ereal-alt-mem*[*of UNIV f y u v*] *assms convex-UNIV v01*
      by (*simp add*: *add.commute*)
    moreover have *f y* < ∞ **using** *m-def y-def* **unfolding** *domain-def* **by** *auto*
    moreover have $*$: *0* < *1* − *ereal v* **using** *v01*
      by (*metis diff-gt-0-iff-gt ereal-less*(*2*) *ereal-minus*(*1*) *one-ereal-def*)
    moreover from $*$ **have** (*1*−*ereal v*) $*$ *f u* = $-\infty$ **using** *u-def* **by** *auto*
    ultimately have *f x* = $-\infty$ **using** *v01* **by** *simp*
  } hence *?thesis* **by** *auto*
} ultimately show *?thesis* **by** *blast*
qed

lemma *convex-improper2*:
  fixes *f* :: *'a*::*euclidean-space* $\Rightarrow$ *ereal*
  assumes *convex-on UNIV f*
  assumes $^\sim$(*proper f*)
  shows *f x* = ∞ | *f x* = $-\infty$ | *x* : *rel-frontier* (*domain f*)
proof−
{ assume *a*: $^\sim$(*f x* = ∞ | *f x* = $-\infty$)

hence *x*: *domain f* **unfolding** *domain-def* **by** *auto*
hence *x* : *closure* (*domain f*) **using** *closure-subset* **by** *auto*
moreover have $x \notin$ *rel-interior* (*domain f*) **using** *assms convex-improper a* **by** *auto*
ultimately have *x* : *rel-frontier* (*domain f*) **unfolding** *rel-frontier-def* **by** *auto*
} **thus** *?thesis* **by** *auto*
qed

lemma *convex-lsc-improper*:
fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes ~(*proper f*)
assumes *lsc f*
shows $f\ x = \infty \mid f\ x = -\infty$
proof−
{ fix *x* **assume** $f\ x \neq \infty$
hence *lsc-at x f* **using** *assms* **unfolding** *lsc-def* **by** *auto*
have *x*: *domain f* **unfolding** *domain-def* **using** ‹$f\ x \neq \infty$› **by** *auto*
hence *x* : *closure* (*domain f*) **using** *closure-subset* **by** *auto*
hence *x-def*: *x* : *closure* (*rel-interior* (*domain f*))
by (*metis assms*(*1*) *convex-closure-rel-interior convex-domain*)
{ fix *C* **assume** *C*<*f x*
from *this* **obtain** *d* **where** *d-def*: $d>0 \wedge (\forall y.\ dist\ x\ y < d \longrightarrow C < f\ y)$
using *lst-at-delta*[*of x f*] ‹*lsc-at x f*› **by** *auto*
from *this* **obtain** *y* **where** *y-def*: *y*:*rel-interior* (*domain f*) ∧ *dist y x* < *d*
using *x-def closure-approachable*[*of x rel-interior* (*domain f*)] **by** *auto*
hence $f\ y = -\infty$ **by** (*metis assms*(*1*) *assms*(*2*) *convex-improper*)
moreover have $C < f\ y$ **using** *y-def d-def* **by** (*simp add*: *dist-commute*)
ultimately have *False* **by** *auto*
} hence $f\ x = -\infty$ **by** *auto*
} from *this* **show** *?thesis* **by** *auto*
qed

lemma *convex-lsc-hull*:
fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
shows *convex-on UNIV* (*lsc-hull f*)
proof−
have *convex*(*Epigraph UNIV f*) **by** (*metis assms convex-EpigraphI convex-UNIV*)
hence *convex* (*Epigraph UNIV* (*lsc-hull f*)) **by** (*metis convex-closure epigraph-lsc-hull*)
thus *?thesis* **by** (*metis convex-Epigraph convex-UNIV*)
qed

lemma *improper-lsc-hull*:
fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
assumes ~(*proper f*)

shows ~(*proper* (*lsc-hull f*))
proof−
{
assume ∗: $\forall x.\ f\ x = \infty$
then have *lsc f*
by (*metis* (*full-types*) *UNIV-I lsc-at-open lsc-def open-UNIV*)
with ∗ **have** $\forall x.$ (*lsc-hull f*) $x = \infty$ **by** (*metis lsc-hull-lsc*)
}
hence $(\forall x.\ f\ x = \infty) \longleftrightarrow (\forall x.$ (*lsc-hull f*) $x = \infty)$
by (*metis ereal-infty-less-eq*(*1*) *lsc-hull-le*)
moreover have $(\exists x.\ f\ x = -\infty) \longrightarrow (\exists x.$ (*lsc-hull f*) $x = -\infty)$
by (*metis ereal-infty-less-eq2*(*2*) *lsc-hull-le*)
ultimately show *?thesis* **using** *assms* **unfolding** *improper-iff* **by** *auto*
qed

lemma *lsc-hull-convex-improper*:
fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes ~(*proper f*)
shows $\forall x \in$ *rel-interior*(*domain f*). (*lsc-hull f*) $x = f\ x$
by (*metis assms convex-improper ereal-infty-less-eq*(*2*) *lsc-hull-le*)

lemma *convex-with-rel-open-domain*:
fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes *rel-open* (*domain f*)
shows $(\forall x.\ f\ x > -\infty) \mid (\forall x.\ (f\ x = \infty \mid f\ x = -\infty))$
proof−
{ **assume** $\neg(\forall x.\ f\ x > -\infty)$
hence ¬(*proper f*) **using** *proper-iff*
by (*simp add*: *proper-iff*)
hence $\forall x \in$ *rel-interior*(*domain f*). $f\ x = -\infty$ **by** (*metis assms*(*1*) *convex-improper*)
hence $\forall x \in$ *domain f*. $f\ x = -\infty$ **by** (*metis assms*(*2*) *rel-open-def*)
hence $\forall x.\ (f\ x = \infty \mid f\ x = -\infty)$ **unfolding** *domain-def* **by** *auto*
} **thus** *?thesis* **by** *auto*
qed

lemma *convex-with-UNIV-domain*:
fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes *domain f* = *UNIV*
shows $(\forall x.\ f\ x > -\infty) \vee (\forall x.\ f\ x = -\infty)$
by (*metis assms convex-improper ereal-MInfty-lessI proper-iff rel-interior-UNIV UNIV-I*)

lemma *rel-interior-Epigraph*:
  fixes f :: $'a$::*euclidean-space* $\Rightarrow$ *ereal*
  assumes *convex-on UNIV f*
  shows (x,z) : *rel-interior* (*Epigraph UNIV f*) $\longleftrightarrow$
    (x : *rel-interior* (*domain f*) $\wedge$ $f\ x <$ *ereal* z)
apply (*subst rel-interior-projection*[*of* - (λy. $\{z.\ (y,\ z)$: *Epigraph UNIV f*$\}$)])
apply (*metis assms convex-EpigraphI convex-UNIV convex-on-ereal-univ*)
unfolding *domain-Epigraph Epigraph-def* **using** *rel-interior-ereal-semiline* **by** *auto*

lemma *rel-interior-EpigraphI*:
  fixes f :: $'a$::*euclidean-space* $\Rightarrow$ *ereal*
  assumes *convex-on UNIV f*
  shows *rel-interior* (*Epigraph UNIV f*) =
    $\{(x,z)\ |x\ z.\ x$: *rel-interior* (*domain f*) $\wedge$ $f\ x <$ *ereal* $z\}$
proof−
{ fix $x\ z$
  have (x,z):*rel-interior* (*Epigraph UNIV f*) $\longleftrightarrow$ (x : *rel-interior* (*domain f*) $\wedge$ f $x <$ *ereal* z)
    using *rel-interior-Epigraph*[*of f x z*] *assms* **by** *auto*
} thus *?thesis* **by** *auto*
qed

lemma *convex-less-ri-domain*:
  fixes f :: $'a$::*euclidean-space* $\Rightarrow$ *ereal*
  assumes *convex-on UNIV f*
  assumes $\exists x.\ f\ x < a$
  shows $\exists x \in$*rel-interior* (*domain f*). $f\ x < a$
proof−
  define A **where** $A = \{(x::'a::$*euclidean-space*$,m)|x\ m.$ *ereal* $m<a\}$
  obtain x **where** $f\ x < a$ **using** *assms* **by** *auto*
  then obtain z **where** *z-def*: $f\ x <$ *ereal* z $\wedge$ *ereal* $z < a$ **using** *ereal-dense2* **by** *auto*
  hence (x,z):A $\wedge$ (x,z):*Epigraph UNIV f* **unfolding** *A-def Epigraph-def* **by** *auto*
  hence A *Int* (*Epigraph UNIV f*) $\neq \{\}$ **unfolding** *A-def Epigraph-def* **using** *assms* **by** *auto*
  moreover have *open A* **proof**(*cases a*)
    case *real* **hence** $A = \{y.$ *inner* $(0::'a,\ 1)\ y <$ *real-of-ereal* $a\}$ **using** *A-def* **by** *auto*
      thus *?thesis* **using** *open-halfspace-lt* **by** *auto*
    next case *PInf* **thus** *?thesis* **using** *A-def* **by** *auto*
    next case *MInf* **thus** *?thesis* **using** *A-def* **by** *auto*
  qed
  ultimately have A *Int rel-interior*(*Epigraph UNIV f*) $\neq \{\}$
    by (*metis assms*(*1*) *convex-Epigraph convex-UNIV*
      open-Int-closure-eq-empty open-inter-closure-rel-interior)
  then obtain $x0\ z0$ **where** $(x0,z0)$:A $\wedge$ $x0$: *rel-interior* (*domain f*) $\wedge$ $f\ x0 <$

ereal z0
  using *rel-interior-Epigraph*[*of f*] *assms* **by** *auto*
 thus *?thesis* **apply**(*rule-tac x=x0* **in** *bexI*) **using** *A-def* **by** *auto*
qed

lemma *rel-interior-eq-between*:
 fixes *S T* :: (*'m*::*euclidean-space*) *set*
 assumes *convex S convex T*
 shows (*rel-interior S* = *rel-interior T*) ⟷ (*rel-interior S* ≤ *T* ∧ *T* ≤ *closure S*)
by (*metis assms closure-eq-between convex-closure-rel-interior convex-rel-interior-closure*)

lemma *convex-less-in-riS*:
 fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*
 assumes *convex-on UNIV f*
 assumes *convex S rel-interior S* ≤ *domain f*
 assumes ∃ *x*∈*closure S*. *f x* < *a*
 shows ∃ *x*∈*rel-interior S*. *f x* < *a*
proof−
 define *g* **where** *g x* = (*if x*:*closure S then* (*f x*) *else* ∞) **for** *x*
 hence ∃ *x*. *g x* < *a* **using** *assms* **by** *auto*
 have *convg*: *convex-on UNIV g* **unfolding** *g-def* **apply** (*subst convex-PInfty-outside*)
  using *assms convex-closure* **by** *auto*
 hence ∗: ∃ *x*∈*rel-interior* (*domain g*). *g x* < *a* **apply** (*subst convex-less-ri-domain*)
  using *assms g-def* **by** *auto*
 have *convex* (*domain g*) **by** (*metis convg convex-domain*)
 moreover have *rel-interior S* ≤ *domain g* ∧ *domain g* ≤ *closure S*
  using *g-def assms rel-interior-subset-closure* **unfolding** *domain-def* **by** *auto*
 ultimately have *rel-interior* (*domain g*) = *rel-interior S*
  by (*metis assms*(*2*) *rel-interior-eq-between*)
 thus *?thesis*
  by (*metis* ∗ *g-def less-ereal.simps*(*2*))
qed

lemma *convex-less-inS*:
 fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*
 assumes *convex-on UNIV f*
 assumes *convex S S* ≤ *domain f*
 assumes ∃ *x*∈*closure S*. *f x* < *a*
 shows ∃ *x*∈*S*. *f x* < *a*
using *convex-less-in-riS*[*of f S a*] *rel-interior-subset*[*of S*] *assms* **by** *auto*

lemma *convex-finite-geq-on-closure*:
 fixes *f* :: *'a*::*euclidean-space* ⇒ *ereal*

assumes *convex-on UNIV f*
assumes *convex S finite-on S f*
assumes $\forall x \in S.\ f\ x \geq a$
shows $\forall x \in closure\ S.\ f\ x \geq a$
proof−
have $S \leq domain\ f$ **using** *assms* **unfolding** *finite-on-def domain-def* **by** *auto*
{ assume $\neg(\forall x \in closure\ S.\ f\ x \geq a)$
hence $\exists x \in closure\ S.\ f\ x < a$ **by** (*simp add*: *not-le*)
hence *False* **using** *convex-less-inS*[*of f S a*] *assms* ‹$S \leq domain\ f$› **by** *auto*
} thus *?thesis* **by** *auto*
qed

lemma *lsc-hull-of-convex-determined*:
fixes *f g* :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f convex-on UNIV g*
assumes *rel-interior* (*domain f*) = *rel-interior* (*domain g*)
assumes $\forall x \in rel\text{-}interior\ (domain\ f).\ f\ x = g\ x$
shows *lsc-hull f* = *lsc-hull g*
proof−
have *rel-interior* (*Epigraph UNIV f*) = *rel-interior* (*Epigraph UNIV g*)
apply (*subst rel-interior-EpigraphI*, *metis assms*)+ **using** *assms* **by** *auto*
hence *closure* (*Epigraph UNIV f*) = *closure* (*Epigraph UNIV g*)
by (*metis assms convex-EpigraphI convex-UNIV convex-closure-rel-interior*)
thus *?thesis* **by** (*metis lsc-hull-expl*)
qed

lemma *domain-lsc-hull-between*:
fixes *f* :: *'a::euclidean-space* ⇒ *ereal*
shows $domain\ f \leq domain\ (lsc\text{-}hull\ f)$
$\wedge\ domain\ (lsc\text{-}hull\ f) \leq closure\ (domain\ f)$
proof−
{ fix *x* **assume** *x*:*domain f*
hence *x*:*domain* (*lsc-hull f*) **unfolding** *domain-def* **using** *lsc-hull-le*[*of f x*] **by** *auto*
} moreover
{ fix *x* **assume** *x*:*domain* (*lsc-hull f*)
hence *min* (*f x*) (*Liminf* (*at x*) *f*) < ∞ **unfolding** *domain-def* **using** *lsc-hull-liminf-at*[*of f*] **by** *auto*
then obtain *z* **where** *z-def*: $min\ (f\ x)\ (Liminf\ (at\ x)\ f) < z \wedge z < \infty$ **by** (*metis dense*)
{ fix *e*::*real* **assume** *e*>*0*
hence $Inf\ (f\ `\ ball\ x\ e) \leq min\ (f\ x)\ (Liminf\ (at\ x)\ f)$
unfolding *min-Liminf-at* **apply** (*subst SUP-upper*) **by** *auto*
hence $\exists y.\ y \in ball\ x\ e \wedge f\ y \leq z$
using *Inf-le-iff-less* [*of f ball x e min* (*f x*) (*Liminf* (*at x*) *f*)] *z-def* **by** (*auto simp add*: *Bex-def*)
hence $\exists y.\ dist\ x\ y < e \wedge y \in domain\ f$ **unfolding** *domain-def ball-def* **using**

z-def **by** *auto*
} **hence** $x \in closure(domain\ f)$ **unfolding** *closure-approachable* **by** (*auto simp add*: *dist-commute*)
} **ultimately show** *?thesis* **by** *auto*
qed

lemma *domain-vs-domain-lsc-hull*:
fixes f :: $'a$::*euclidean-space* $\Rightarrow$ *ereal*
assumes *convex-on UNIV f*
shows *rel-interior*(*domain* (*lsc-hull f*)) = *rel-interior*(*domain f*)
$\wedge$ *closure*(*domain* (*lsc-hull f*)) = *closure*(*domain f*)
$\wedge$ *aff-dim*(*domain* (*lsc-hull f*)) = *aff-dim*(*domain f*)
proof−
have *convex* (*domain f*) **by** (*metis assms convex-domain*)
moreover have *convex* (*domain* (*lsc-hull f*)) **by** (*metis assms convex-domain convex-lsc-hull*)
moreover have *rel-interior* (*domain f*) $\leq$ *domain* (*lsc-hull f*)
$\wedge$ *domain* (*lsc-hull f*) $\leq$ *closure* (*domain f*)
by (*metis domain-lsc-hull-between rel-interior-subset subset-trans*)
ultimately show *?thesis* **by** (*metis closure-eq-between rel-interior-aff-dim rel-interior-eq-between*)
qed

lemma *vertical-line-affine*:
fixes x :: $'a$::*euclidean-space*
shows *affine* $\{(x,m::real)|m.\ m{:}UNIV\}$
unfolding *affine-def* **by** (*auto simp add*: *pth-8*)

lemma *lsc-hull-of-convex-agrees-onRI*:
fixes f :: $'a$::*euclidean-space* $\Rightarrow$ *ereal*
assumes *convex-on UNIV f*
shows $\forall x \in$ *rel-interior* (*domain f*). ($f\ x$ = (*lsc-hull f*) x)
proof−
have *cEpi*: *convex* (*Epigraph UNIV f*) **by** (*metis assms convex-EpigraphI convex-UNIV*)
{ **fix** x **assume** *x-def*: x : *rel-interior* (*domain f*)
hence $f\ x < \infty$ **unfolding** *domain-def* **using** *rel-interior-subset* **by** *auto*
then obtain r **where** *r-def*: (x,r):*rel-interior* (*Epigraph UNIV f*)
using *assms x-def rel-interior-Epigraph*[*of f x*] **by** (*metis ereal-dense2*)
define M **where** $M = \{(x,m::real)|m.\ m{:}UNIV\}$
hence *affine M* **using** *vertical-line-affine* **by** *auto*
moreover have *rel-interior* (*Epigraph UNIV f*) *Int M* $\neq \{\}$ **using** *r-def M-def* **by** *auto*
ultimately have $*$: *closure* (*Epigraph UNIV f*) *Int M* = *closure* (*Epigraph UNIV f Int M*)
using *convex-affine-closure-Int*[*of Epigraph UNIV f M*] *cEpi* **by** *auto*
have *Epigraph UNIV f Int M* = $\{x\} \times \{m.\ f\ x \leq ereal\ m\}$

unfolding *Epigraph-def M-def* **by** *auto*
moreover have *closed*$(\{x\} \times \{m.\ f\ x \leq\ ereal\ m\})$ **apply** (*subst closed-Times*)
using *closed-ereal-semiline* **by** *auto*
ultimately have $\{x\} \times \{m.\ f\ x \leq ereal\ m\} = closure\ (Epigraph\ UNIV\ f)\ Int\ M$
by (*metis* $*$ *Int-commute closure-closed*)
also have $\ldots$=*Epigraph UNIV* (*lsc-hull f*) *Int M* **by** (*metis Int-commute epigraph-lsc-hull*)
also have $\ldots=\{x\} \times \{m.\ ((lsc\text{-}hull\ f)\ x) \leq ereal\ m\}$
unfolding *Epigraph-def M-def* **by** *auto*
finally have $\{m.\ f\ x \leq ereal\ m\} = \{m.\ lsc\text{-}hull\ f\ x \leq ereal\ m\}$ **by** *auto*
hence $f\ x = (lsc\text{-}hull\ f)\ x$ **using** *ereal-semiline-unique* **by** *auto*
} **thus** *?thesis* **by** *auto*
qed

lemma *lsc-hull-of-convex-agrees-outside*:
fixes $f :: 'a::euclidean\text{-}space \Rightarrow ereal$
assumes *convex-on UNIV f*
shows $\forall x.\ x \notin closure\ (domain\ f) \longrightarrow (f\ x = (lsc\text{-}hull\ f)\ x)$
proof−
{ **fix** x **assume** $x \notin closure\ (domain\ f)$
hence $x \notin domain\ (lsc\text{-}hull\ f)$ **using** *domain-lsc-hull-between* **by** *auto*
hence $(lsc\text{-}hull\ f)\ x = \infty$ **unfolding** *domain-def* **by** *auto*
hence $f\ x = (lsc\text{-}hull\ f)\ x$ **using** *lsc-hull-le*[*of f x*] **by** *auto*
} **thus** *?thesis* **by** *auto*
qed

lemma *lsc-hull-of-convex-agrees*:
fixes $f :: 'a::euclidean\text{-}space \Rightarrow ereal$
assumes *convex-on UNIV f*
shows $\forall x.\ (f\ x = (lsc\text{-}hull\ f)\ x) \mid x : rel\text{-}frontier\ (domain\ f)$
by (*metis DiffI assms lsc-hull-of-convex-agrees-onRI lsc-hull-of-convex-agrees-outside rel-frontier-def*)

lemma *lsc-hull-of-proper-convex-proper*:
fixes $f :: 'a::euclidean\text{-}space \Rightarrow ereal$
assumes *convex-on UNIV f proper f*
shows *proper* (*lsc-hull f*)
proof−
obtain x **where** *x-def*: $x{:}rel\text{-}interior\ (domain\ f) \wedge f\ x < \infty$
by (*metis assms convex-less-ri-domain ereal-less-PInfty proper-iff*)
hence $f\ x = (lsc\text{-}hull\ f)\ x$ **using** *lsc-hull-of-convex-agrees*[*of f*] *assms*
unfolding *rel-frontier-def* **by** *auto*
moreover have $f\ x > -\infty$
using *assms proper-iff* **by** *blast*
ultimately have $(lsc\text{-}hull\ f)\ x < \infty \wedge (lsc\text{-}hull\ f)\ x > -\infty$ **using** *x-def* **by** *auto*
thus *?thesis* **using** *convex-lsc-improper*[*of lsc-hull f x*]

lsc-lsc-hull[*of f*] *assms convex-lsc-hull*[*of f*] **by** *auto*
qed

lemma *lsc-hull-of-proper-convex*:
fixes f :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f proper f*
shows *lsc* (*lsc-hull f*) ∧ *proper* (*lsc-hull f*) ∧ *convex-on UNIV* (*lsc-hull f*) ∧
(∀ *x*. (*f x* = (*lsc-hull f*) *x*) | *x* : *rel-frontier* (*domain f*))
by (*metis assms convex-lsc-hull lsc-hull-of-convex-agrees lsc-hull-of-proper-convex-proper lsc-lsc-hull*)

lemma *affine-no-rel-frontier*:
fixes *S* :: (*'n::euclidean-space*) *set*
assumes *affine S*
shows *rel-frontier S* = {}
unfolding *rel-frontier-def* **using** *assms affine-closed*[*of S*]
closure-closed[*of S*] *affine-rel-open*[*of S*] *rel-open-def*[*of S*] **by** *auto*

lemma *convex-with-affine-domain-is-lsc*:
fixes f :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes *affine* (*domain f*)
shows *lsc f*
by (*metis assms affine-no-rel-frontier emptyE lsc-def lsc-hull-liminf-at*
lsc-hull-of-convex-agrees lsc-liminf-at-eq)

lemma *convex-finite-is-lsc*:
fixes f :: *'a::euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes *finite-on UNIV f*
shows *lsc f*
proof−
have *affine* (*domain f*)
using *assms affine-UNIV* **unfolding** *finite-on-def domain-def* **by** *auto*
thus *?thesis* **by** (*metis assms*(*1*) *convex-with-affine-domain-is-lsc*)
qed

lemma *always-eventually-within*:
(∀ *x*∈*S*. *P x*) ⟹ *eventually P* (*at x within S*)
unfolding *eventually-at-filter* **by** *auto*

lemma *ereal-divide-pos*:
assumes (*a::ereal*)>*0 b*>*0*

shows *a/(ereal b)>0*
by (*metis PInfty-eq-infinity assms ereal.simps*(*2*) *ereal-less*(*2*) *ereal-less-divide-pos ereal-mult-zero*)

lemma *real-interval-limpt*:
assumes *a<b*
shows (*b*::*real*) *islimpt* {*a..<b*}
proof−
{ **fix** *T* **assume** *b*:*T open T*
then obtain *e* **where** *e-def*: *e>0* ∧ *cball b e* ≤ *T* **using** *open-contains-cball*[*of T*] **by** *auto*
hence (*b*−*e*):*cball b e* **unfolding** *cball-def dist-norm* **by** *auto*
moreover
{ **assume** *a*≥*b*−*e* **hence** *a*:*cball b e* **unfolding** *cball-def dist-norm* **using** ‹*a<b*› **by** *auto* }
ultimately have *max a* (*b*−*e*):*cball b e*
by (*metis max.absorb1 max.absorb2 linear*)
hence *max a* (*b*−*e*):*T* **using** *e-def* **by** *auto*
moreover have *max a* (*b*−*e*):{*a..<b*} **using** *e-def* ‹*a<b*› **by** *auto*
ultimately have ∃ *y*∈{*a..<b*}. *y* : *T* ∧ *y* ≠ *b* **by** *auto*
} **thus** *?thesis* **unfolding** *islimpt-def* **by** *auto*
qed

lemma *lsc-hull-of-convex-aux*:
Limsup (*at 1 within* {*0..<1*}) (λ*m. ereal* ((*1*−*m*)∗*a*+*m*∗*b*)) ≤ *ereal b*
proof−
have *nontr*: ~*trivial-limit* (*at 1 within* {*0..< 1*::*real*})
apply (*subst trivial-limit-within*) **using** *real-interval-limpt* **by** *auto*
have ((λ*m. ereal* ((*1*−*m*)∗*a*+*m*∗*b*)) ⟶ (*1* − *1*) ∗ *a* + *1* ∗ *b*) (*at 1 within* {*0..<1*})
unfolding *lim-ereal* **by** (*intro tendsto-intros*)
from *lim-imp-Limsup*[*OF nontr this*] **show** *?thesis* **by** *simp*
qed

lemma *lsc-hull-of-convex*:
fixes *f* :: ′*a*::*euclidean-space* ⇒ *ereal*
assumes *convex-on UNIV f*
assumes *x* : *rel-interior* (*domain f*)
shows ((λ*m. f*((*1*−*m*)$*_R$ *x* + *m* $*_R$ *y*)) ⟶ (*lsc-hull f*) *y*) (*at 1 within* {*0..<1*})
(**is** (*?g* ⟶ - *y*) -)
proof (*cases y*=*x*)
case *True*
hence *?g* = (λ*m. f y*) **by** (*simp add*: *algebra-simps*)
hence (*?g* ⟶ *f y*) (*at 1 within* {*0..<1*}) **by** *simp*
moreover have (*lsc-hull f*) *y* = *f y* **by** (*metis* ‹*y*=*x*› *assms lsc-hull-of-convex-agrees-onRI*)
ultimately show *?thesis* **by** *auto*

next
case *False*
have *aux*: $\forall m.\ y - ((1 - m) *_R x + m *_R y) = (1-m) *_R (y-x)$ **by** (*simp add*: *algebra-simps*)
have (*lsc-hull f*) $y = min\ (f\ y)$ (*Liminf* (*at y*) *f*) **by** (*metis lsc-hull-liminf-at*)
also have $\ldots \leq$ *Liminf* (*at 1 within* $\{0..<1\}$) *?g* **unfolding** *min-Liminf-at* **unfolding** *Liminf-within*
apply (*subst SUP-mono*) **apply** (*rule-tac* $x=n/norm(y-x)$ **in** *bexI*)
apply (*subst INF-mono*) **apply** (*rule-tac* $x=(1 - m) *_R x + m *_R y$ **in** *bexI*)
prefer *2*
unfolding *ball-def dist-norm* **by** (*auto simp add*: *aux* ‹$y \neq x$› *less-divide-eq*)
finally have $*$: (*lsc-hull f*) $y \leq$ *Liminf* (*at 1 within* $\{0..<1\}$) *?g* **by** *auto*
{ fix *b* **assume** *ereal* $b \geq$(*lsc-hull f*) *y*
hence *yb*: (y,b):*closure*(*Epigraph UNIV f*) **by** (*metis epigraph-lsc-hull mem-Epigraph UNIV-I*)
have *x* : *domain f* **by** (*metis assms*(*2*) *rel-interior-subset rev-subsetD*)
hence $f\ x<\infty$ **unfolding** *domain-def* **by** *auto*
then obtain *a* **where***ereal* $a>f\ x$ **by** (*metis ereal-dense2*)
hence *xa*: (x,a):*rel-interior*(*Epigraph UNIV f*) **by** (*metis assms rel-interior-Epigraph*)
{ fix *m* :: *real* **assume** $0 \leq m \wedge m < 1$
hence $(y, b) - (1-m) *_R ((y, b) - (x, a))$: *rel-interior* (*Epigraph UNIV f*)
apply (*subst rel-interior-closure-convex-shrink*)
apply (*metis assms*(*1*) *convex-Epigraph convex-UNIV convex-on-ereal-univ*)
using *yb xa* **by** *auto*
hence $f\ (y - (1-m) *_R (y-x)\ ) <$ *ereal* $(b-(1-m)*(b-a))$
using *assms*(*1*) *rel-interior-Epigraph* **by** *auto*
hence *?g* $m \leq$ *ereal* $((1-m)*a+m*b)$ **by** (*simp add*: *algebra-simps*)
}
hence *eventually* ($\lambda m.$ *?g* $m \leq$ *ereal* $((1-m)*a+m*b)$)
(*at 1 within* $\{0..<1\}$) **apply** (*subst always-eventually-within*) **by** *auto*
hence *Limsup* (*at 1 within* $\{0..<1\}$) *?g* $\leq$ *Limsup* (*at 1 within* $\{0..<1\}$) ($\lambda m.$ *ereal* $((1-m)*a+m*b)$)
apply (*subst Limsup-mono*) **by** *auto*
also have $\ldots \leq$ *ereal b* **using** *lsc-hull-of-convex-aux* **by** *auto*
finally have *Limsup* (*at 1 within* $\{0..<1\}$) *?g* $\leq$*ereal b* **by** *auto*
}
hence *Limsup* (*at 1 within* $\{0..<1\}$) *?g* $\leq$ (*lsc-hull f*) *y*
using *ereal-le-real*[*of* (*lsc-hull f*) *y*] **by** *auto*
moreover have *nontr*: ~*trivial-limit* (*at* (*1*::*real*) *within* $\{0..<1\}$)
apply (*subst trivial-limit-within*) **using** *real-interval-limpt* **by** *auto*
moreover hence *Liminf* (*at 1 within* $\{0..<1\}$) *?g* $\leq$ *Limsup* (*at 1 within* $\{0..<1\}$) *?g*
apply (*subst Liminf-le-Limsup*) **by** *auto*
ultimately have *Limsup* (*at 1 within* $\{0..<1\}$) *?g* $=$ (*lsc-hull f*) *y*
$\wedge$ *Liminf* (*at 1 within* $\{0..<1\}$) *?g* $=$ (*lsc-hull f*) *y*
using $*$ **by** *auto*
thus *?thesis* **apply** (*subst Liminf-eq-Limsup*) **using** *nontr* **by** *auto*
qed

end