

The Hahn-Banach Theorem for Real Vector Spaces

Gertrud Bauer

<http://www.in.tum.de/~bauerg/>

April 19, 2009

Abstract

The Hahn-Banach Theorem is one of the most fundamental results in functional analysis. We present a fully formal proof of two versions of the theorem, one for general linear spaces and another for normed spaces. This development is based on simply-typed classical set-theory, as provided by Isabelle/HOL.

Contents

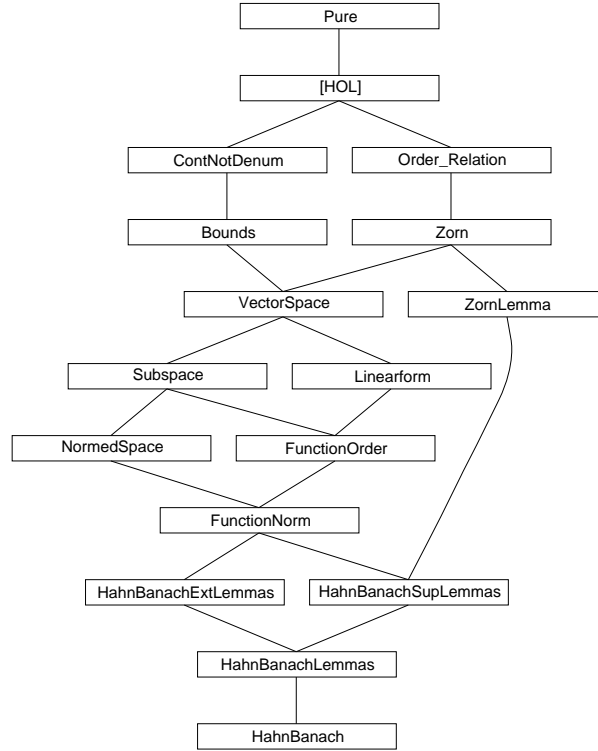
1	Preface	3
I	Basic Notions	4
2	Bounds	4
3	Vector spaces	5
3.1	Signature	5
3.2	Vector space laws	6
4	Subspaces	13
4.1	Definition	13
4.2	Linear closure	15
4.3	Sum of two vectorspaces	16
4.4	Direct sums	18
5	Normed vector spaces	21
5.1	Quasinorms	22
5.2	Norms	22
5.3	Normed vector spaces	23
6	Linearforms	23

7	An order on functions	24
7.1	The graph of a function	24
7.2	Functions ordered by domain extension	25
7.3	Domain and function of a graph	25
7.4	Norm-preserving extensions of a function	26
8	The norm of a function	27
8.1	Continuous linear forms	27
8.2	The norm of a linear form	27
9	Zorn's Lemma	31
II	Lemmas for the Proof	33
10	The supremum w.r.t. the function order	33
11	Extending non-maximal functions	40
III	The Main Proof	46
12	The Hahn-Banach Theorem	46
12.1	The Hahn-Banach Theorem for vector spaces	46
12.2	Alternative formulation	51
12.3	The Hahn-Banach Theorem for normed spaces	51

1 Preface

This is a fully formal proof of the Hahn-Banach Theorem. It closely follows the informal presentation given in Heuser's textbook [1, § 36]. Another formal proof of the same theorem has been done in Mizar [3]. A general overview of the relevance and history of the Hahn-Banach Theorem is given by Narici and Beckenstein [2].

The document is structured as follows. The first part contains definitions of basic notions of linear algebra: vector spaces, subspaces, normed spaces, continuous linear-forms, norm of functions and an order on functions by domain extension. The second part contains some lemmas about the supremum (w.r.t. the function order) and extension of non-maximal functions. With these preliminaries, the main proof of the theorem (in its two versions) is conducted in the third part. The dependencies of individual theories are as follows.



2 Bounds

```

lemma the-lubI-ex:
  assumes ex:  $\exists x. \text{lub } A \ x$ 

```

```

  shows lub A ( $\sqcup$  A)
proof -
  from ex obtain x where x: lub A x ..
  also from x have [symmetric]:  $\sqcup$  A = x ..
  finally show ?thesis .
qed

lemma lub-compat: lub A x = isLub UNIV A x
proof -
  have isUb UNIV A = ( $\lambda x. A * \leq x \wedge x \in UNIV$ )
  by (rule ext) (simp only: isUb-def)
  then show ?thesis
  by (simp only: lub-def isLub-def leastP-def setge-def settle-def) blast
qed

lemma real-complete:
  fixes A :: real set
  assumes nonempty:  $\exists a. a \in A$ 
  and ex-upper:  $\exists y. \forall a \in A. a \leq y$ 
  shows  $\exists x. \text{lub } A \ x$ 
proof -
  from ex-upper have  $\exists y. \text{isUb } UNIV \ A \ y$ 
  unfolding isUb-def settle-def by blast
  with nonempty have  $\exists x. \text{isLub } UNIV \ A \ x$ 
  by (rule reals-complete)
  then show ?thesis by (simp only: lub-compat)
qed

end

```

3 Vector spaces

```

theory VectorSpace
imports Real Bounds Zorn
begin

```

3.1 Signature

For the definition of real vector spaces a type $'a$ of the sort $\{plus, minus, zero\}$ is considered, on which a real scalar multiplication \cdot is declared.

```

consts
  prod :: real  $\Rightarrow$   $'a::\{plus, minus, zero\} \Rightarrow 'a$     (infixr  $'(*)$  70)

notation (xsymbols)
  prod (infixr  $\cdot$  70)
notation (HTML output)
  prod (infixr  $\cdot$  70)

```

3.2 Vector space laws

A *vector space* is a non-empty set V of elements from $'a$ with the following vector space laws: The set V is closed under addition and scalar multiplication, addition is associative and commutative; $-x$ is the inverse of x w. r. t. addition and 0 is the neutral element of addition. Addition and multiplication are distributive; scalar multiplication is associative and the real number 1 is the neutral element of scalar multiplication.

locale *var-V* = **fixes** V

locale *vectorspace* = *var-V* +

assumes *non-empty* [*iff*, *intro?*]: $V \neq \{\}$
and *add-closed* [*iff*]: $x \in V \implies y \in V \implies x + y \in V$
and *mult-closed* [*iff*]: $x \in V \implies a \cdot x \in V$
and *add-assoc*: $x \in V \implies y \in V \implies z \in V \implies (x + y) + z = x + (y + z)$
and *add-commute*: $x \in V \implies y \in V \implies x + y = y + x$
and *diff-self* [*simp*]: $x \in V \implies x - x = 0$
and *add-zero-left* [*simp*]: $x \in V \implies 0 + x = x$
and *add-mult-distrib1*: $x \in V \implies y \in V \implies a \cdot (x + y) = a \cdot x + a \cdot y$
and *add-mult-distrib2*: $x \in V \implies (a + b) \cdot x = a \cdot x + b \cdot x$
and *mult-assoc*: $x \in V \implies (a * b) \cdot x = a \cdot (b \cdot x)$
and *mult-1* [*simp*]: $x \in V \implies 1 \cdot x = x$
and *negate-eq1*: $x \in V \implies -x = (-1) \cdot x$
and *diff-eq1*: $x \in V \implies y \in V \implies x - y = x + -y$

lemma (**in** *vectorspace*) *negate-eq2*: $x \in V \implies (-1) \cdot x = -x$
by (*rule negate-eq1* [*symmetric*])

lemma (**in** *vectorspace*) *negate-eq2a*: $x \in V \implies -1 \cdot x = -x$
by (*simp add: negate-eq1*)

lemma (**in** *vectorspace*) *diff-eq2*: $x \in V \implies y \in V \implies x + -y = x - y$
by (*rule diff-eq1* [*symmetric*])

lemma (**in** *vectorspace*) *diff-closed* [*iff*]: $x \in V \implies y \in V \implies x - y \in V$
by (*simp add: diff-eq1 negate-eq1*)

lemma (**in** *vectorspace*) *neg-closed* [*iff*]: $x \in V \implies -x \in V$
by (*simp add: negate-eq1*)

lemma (**in** *vectorspace*) *add-left-commute*:

$x \in V \implies y \in V \implies z \in V \implies x + (y + z) = y + (x + z)$

proof –

assume *xyz*: $x \in V \ y \in V \ z \in V$

then have $x + (y + z) = (x + y) + z$

by (*simp only: add-assoc*)

also from *xyz* **have** $\dots = (y + x) + z$ **by** (*simp only: add-commute*)

also from *xyz* **have** $\dots = y + (x + z)$ **by** (*simp only: add-assoc*)

finally show *?thesis* .

qed

theorems (**in** *vectorspace*) *add-ac* =

add-assoc add-commute add-left-commute

The existence of the zero element of a vector space follows from the non-emptiness of carrier set.

```
lemma (in vectorspace) zero [iff]:  $0 \in V$ 
proof -
  from non-empty obtain x where  $x: x \in V$  by blast
  then have  $0 = x - x$  by (rule diff-self [symmetric])
  also from x x have  $\dots \in V$  by (rule diff-closed)
  finally show ?thesis .
qed
```

```
lemma (in vectorspace) add-zero-right [simp]:
 $x \in V \implies x + 0 = x$ 
proof -
  assume  $x: x \in V$ 
  from this and zero have  $x + 0 = 0 + x$  by (rule add-commute)
  also from x have  $\dots = x$  by (rule add-zero-left)
  finally show ?thesis .
qed
```

```
lemma (in vectorspace) mult-assoc2:
 $x \in V \implies a \cdot b \cdot x = (a * b) \cdot x$ 
by (simp only: mult-assoc)
```

```
lemma (in vectorspace) diff-mult-distrib1:
 $x \in V \implies y \in V \implies a \cdot (x - y) = a \cdot x - a \cdot y$ 
by (simp add: diff-eq1 negate-eq1 add-mult-distrib1 mult-assoc2)
```

```
lemma (in vectorspace) diff-mult-distrib2:
 $x \in V \implies (a - b) \cdot x = a \cdot x - (b \cdot x)$ 
proof -
  assume  $x: x \in V$ 
  have  $(a - b) \cdot x = (a + - b) \cdot x$ 
  by (simp add: real-diff-def)
  also from x have  $\dots = a \cdot x + (- b) \cdot x$ 
  by (rule add-mult-distrib2)
  also from x have  $\dots = a \cdot x + - (b \cdot x)$ 
  by (simp add: negate-eq1 mult-assoc2)
  also from x have  $\dots = a \cdot x - (b \cdot x)$ 
  by (simp add: diff-eq1)
  finally show ?thesis .
qed
```

```
lemmas (in vectorspace) distrib =
  add-mult-distrib1 add-mult-distrib2
  diff-mult-distrib1 diff-mult-distrib2
```

Further derived laws:

```
lemma (in vectorspace) mult-zero-left [simp]:
 $x \in V \implies 0 \cdot x = 0$ 
proof -
  assume  $x: x \in V$ 
  have  $0 \cdot x = (1 - 1) \cdot x$  by simp
  also have  $\dots = (1 + - 1) \cdot x$  by simp
```

```

also from  $x$  have ... =  $1 \cdot x + (-1) \cdot x$ 
  by (rule add-mult-distrib2)
also from  $x$  have ... =  $x + (-1) \cdot x$  by simp
also from  $x$  have ... =  $x + -x$  by (simp add: negate-eq2a)
also from  $x$  have ... =  $x - x$  by (simp add: diff-eq2)
also from  $x$  have ... =  $0$  by simp
finally show ?thesis .
qed

```

```

lemma (in vectorspace) mult-zero-right [simp]:
   $a \cdot 0 = (0::'a)$ 
proof -
  have  $a \cdot 0 = a \cdot (0 - (0::'a))$  by simp
  also have ... =  $a \cdot 0 - a \cdot 0$ 
    by (rule diff-mult-distrib1) simp-all
  also have ... =  $0$  by simp
  finally show ?thesis .
qed

```

```

lemma (in vectorspace) minus-mult-cancel [simp]:
   $x \in V \implies (-a) \cdot -x = a \cdot x$ 
  by (simp add: negate-eq1 mult-assoc2)

```

```

lemma (in vectorspace) add-minus-left-eq-diff:
   $x \in V \implies y \in V \implies -x + y = y - x$ 
proof -
  assume  $xy: x \in V \ y \in V$ 
  then have  $-x + y = y + -x$  by (simp add: add-commute)
  also from  $xy$  have ... =  $y - x$  by (simp add: diff-eq1)
  finally show ?thesis .
qed

```

```

lemma (in vectorspace) add-minus [simp]:
   $x \in V \implies x + -x = 0$ 
  by (simp add: diff-eq2)

```

```

lemma (in vectorspace) add-minus-left [simp]:
   $x \in V \implies -x + x = 0$ 
  by (simp add: diff-eq2 add-commute)

```

```

lemma (in vectorspace) minus-minus [simp]:
   $x \in V \implies -(-x) = x$ 
  by (simp add: negate-eq1 mult-assoc2)

```

```

lemma (in vectorspace) minus-zero [simp]:
   $-(0::'a) = 0$ 
  by (simp add: negate-eq1)

```

```

lemma (in vectorspace) minus-zero-iff [simp]:
   $x \in V \implies (-x = 0) = (x = 0)$ 
proof
  assume  $x: x \in V$ 
  {
    from  $x$  have  $x = -(-x)$  by (simp add: minus-minus)
  }

```



```

    also assume  $-x = 0$ 
    also have  $-\dots = 0$  by (rule minus-zero)
    finally show  $x = 0$  .
  next
    assume  $x = 0$ 
    then show  $-x = 0$  by simp
  }
qed

```

```

lemma (in vectorspace) add-minus-cancel [simp]:
   $x \in V \implies y \in V \implies x + (-x + y) = y$ 
  by (simp add: add-assoc [symmetric] del: add-commute)

```

```

lemma (in vectorspace) minus-add-cancel [simp]:
   $x \in V \implies y \in V \implies -x + (x + y) = y$ 
  by (simp add: add-assoc [symmetric] del: add-commute)

```

```

lemma (in vectorspace) minus-add-distrib [simp]:
   $x \in V \implies y \in V \implies -(x + y) = -x + -y$ 
  by (simp add: negate-eq1 add-mult-distrib1)

```

```

lemma (in vectorspace) diff-zero [simp]:
   $x \in V \implies x - 0 = x$ 
  by (simp add: diff-eq1)

```

```

lemma (in vectorspace) diff-zero-right [simp]:
   $x \in V \implies 0 - x = -x$ 
  by (simp add: diff-eq1)

```

```

lemma (in vectorspace) add-left-cancel:
   $x \in V \implies y \in V \implies z \in V \implies (x + y = x + z) = (y = z)$ 

```

```

proof
  assume  $x: x \in V$  and  $y: y \in V$  and  $z: z \in V$ 
  {
    from  $y$  have  $y = 0 + y$  by simp
    also from  $x y$  have  $\dots = (-x + x) + y$  by simp
    also from  $x y$  have  $\dots = -x + (x + y)$ 
      by (simp add: add-assoc neg-closed)
    also assume  $x + y = x + z$ 
    also from  $x z$  have  $-x + (x + z) = -x + x + z$ 
      by (simp add: add-assoc [symmetric] neg-closed)
    also from  $x z$  have  $\dots = z$  by simp
    finally show  $y = z$  .
  }
next
  assume  $y = z$ 
  then show  $x + y = x + z$  by (simp only:)
}
qed

```

```

lemma (in vectorspace) add-right-cancel:
   $x \in V \implies y \in V \implies z \in V \implies (y + x = z + x) = (y = z)$ 
  by (simp only: add-commute add-left-cancel)

```

```

lemma (in vectorspace) add-associative:

```

$x \in V \implies y \in V \implies x' \in V \implies y' \in V \implies z \in V$
 $\implies x + y = x' + y' \implies x + (y + z) = x' + (y' + z)$
by (*simp only: add-assoc [symmetric]*)

lemma (*in vectorspace*) *mult-left-commute*:

$x \in V \implies a \cdot b \cdot x = b \cdot a \cdot x$

by (*simp add: real-mult-commute mult-assoc2*)

lemma (*in vectorspace*) *mult-zero-uniq*:

$x \in V \implies x \neq 0 \implies a \cdot x = 0 \implies a = 0$

proof (*rule classical*)

assume $a: a \neq 0$

assume $x: x \in V \ x \neq 0$ **and** $ax: a \cdot x = 0$

from $x \ a$ **have** $x = (\text{inverse } a * a) \cdot x$ **by** *simp*

also from $\langle x \in V \rangle$ **have** $\dots = \text{inverse } a \cdot (a \cdot x)$ **by** (*rule mult-assoc*)

also from ax **have** $\dots = \text{inverse } a \cdot 0$ **by** *simp*

also have $\dots = 0$ **by** *simp*

finally have $x = 0$.

with $\langle x \neq 0 \rangle$ **show** $a = 0$ **by** *contradiction*

qed

lemma (*in vectorspace*) *mult-left-cancel*:

$x \in V \implies y \in V \implies a \neq 0 \implies (a \cdot x = a \cdot y) = (x = y)$

proof

assume $x: x \in V$ **and** $y: y \in V$ **and** $a: a \neq 0$

from x **have** $x = 1 \cdot x$ **by** *simp*

also from a **have** $\dots = (\text{inverse } a * a) \cdot x$ **by** *simp*

also from x **have** $\dots = \text{inverse } a \cdot (a \cdot x)$

by (*simp only: mult-assoc*)

also assume $a \cdot x = a \cdot y$

also from $a \ y$ **have** $\text{inverse } a \cdot \dots = y$

by (*simp add: mult-assoc2*)

finally show $x = y$.

next

assume $x = y$

then show $a \cdot x = a \cdot y$ **by** (*simp only:*)

qed

lemma (*in vectorspace*) *mult-right-cancel*:

$x \in V \implies x \neq 0 \implies (a \cdot x = b \cdot x) = (a = b)$

proof

assume $x: x \in V$ **and** $neg: x \neq 0$

{

from x **have** $(a - b) \cdot x = a \cdot x - b \cdot x$

by (*simp add: diff-mult-distrib2*)

also assume $a \cdot x = b \cdot x$

with x **have** $a \cdot x - b \cdot x = 0$ **by** *simp*

finally have $(a - b) \cdot x = 0$.

with $x \ neg$ **have** $a - b = 0$ **by** (*rule mult-zero-uniq*)

then show $a = b$ **by** *simp*

next

assume $a = b$

then show $a \cdot x = b \cdot x$ **by** (*simp only:*)

}

qed

lemma (in *vectorspace*) *eq-diff-eq*:

$$x \in V \implies y \in V \implies z \in V \implies (x = z - y) = (x + y = z)$$

proof

assume $x: x \in V$ and $y: y \in V$ and $z: z \in V$

{
 assume $x = z - y$
 then have $x + y = z - y + y$ by *simp*
 also from $y z$ have $\dots = z + - y + y$
 by (*simp add: diff-eq1*)
 also have $\dots = z + (- y + y)$
 by (*rule add-assoc*) (*simp-all add: y z*)
 also from $y z$ have $\dots = z + 0$
 by (*simp only: add-minus-left*)
 also from z have $\dots = z$
 by (*simp only: add-zero-right*)
 finally show $x + y = z$.

next

assume $x + y = z$
 then have $z - y = (x + y) - y$ by *simp*
 also from $x y$ have $\dots = x + y + - y$
 by (*simp add: diff-eq1*)
 also have $\dots = x + (y + - y)$
 by (*rule add-assoc*) (*simp-all add: x y*)
 also from $x y$ have $\dots = x$ by *simp*
 finally show $x = z - y$..

}

qed

lemma (in *vectorspace*) *add-minus-eq-minus*:

$$x \in V \implies y \in V \implies x + y = 0 \implies x = - y$$

proof -

assume $x: x \in V$ and $y: y \in V$
 from $x y$ have $x = (- y + y) + x$ by *simp*
 also from $x y$ have $\dots = - y + (x + y)$ by (*simp add: add-ac*)
 also assume $x + y = 0$
 also from y have $- y + 0 = - y$ by *simp*
 finally show $x = - y$.

qed

lemma (in *vectorspace*) *add-minus-eq*:

$$x \in V \implies y \in V \implies x - y = 0 \implies x = y$$

proof -

assume $x: x \in V$ and $y: y \in V$
 assume $x - y = 0$
 with $x y$ have *eq*: $x + - y = 0$ by (*simp add: diff-eq1*)
 with - - have $x = - (- y)$
 by (*rule add-minus-eq-minus*) (*simp-all add: x y*)
 with $x y$ show $x = y$ by *simp*

qed

lemma (in *vectorspace*) *add-diff-swap*:

$$a \in V \implies b \in V \implies c \in V \implies d \in V \implies a + b = c + d$$

$\implies a - c = d - b$
proof –
 assume $vs: a \in V \ b \in V \ c \in V \ d \in V$
 and $eq: a + b = c + d$
 then have $-c + (a + b) = -c + (c + d)$
 by $(simp \ add: \ add\text{-left-cancel})$
 also have $\dots = d$ using $\langle c \in V \rangle \langle d \in V \rangle$ by $(rule \ minus\text{-add-cancel})$
 finally have $eq: -c + (a + b) = d$.
 from vs have $a - c = (-c + (a + b)) + -b$
 by $(simp \ add: \ add\text{-ac} \ diff\text{-eq1})$
 also from $vs \ eq$ have $\dots = d + -b$
 by $(simp \ add: \ add\text{-right-cancel})$
 also from vs have $\dots = d - b$ by $(simp \ add: \ diff\text{-eq2})$
 finally show $a - c = d - b$.
qed

lemma (in *vectorspace*) *vs-add-cancel-21*:
 $x \in V \implies y \in V \implies z \in V \implies u \in V$
 $\implies (x + (y + z) = y + u) = (x + z = u)$
proof
 assume $vs: x \in V \ y \in V \ z \in V \ u \in V$
 {
 from vs have $x + z = -y + y + (x + z)$ by *simp*
 also have $\dots = -y + (y + (x + z))$
 by $(rule \ add\text{-assoc}) \ (simp\text{-all} \ add: \ vs)$
 also from vs have $y + (x + z) = x + (y + z)$
 by $(simp \ add: \ add\text{-ac})$
 also assume $x + (y + z) = y + u$
 also from vs have $-y + (y + u) = u$ by *simp*
 finally show $x + z = u$.
 next
 assume $x + z = u$
 with vs show $x + (y + z) = y + u$
 by $(simp \ only: \ add\text{-left-commute} \ [of \ x])$
 }
qed

lemma (in *vectorspace*) *add-cancel-end*:
 $x \in V \implies y \in V \implies z \in V \implies (x + (y + z) = y) = (x = -z)$
proof
 assume $vs: x \in V \ y \in V \ z \in V$
 {
 assume $x + (y + z) = y$
 with vs have $(x + z) + y = 0 + y$
 by $(simp \ add: \ add\text{-ac})$
 with vs have $x + z = 0$
 by $(simp \ only: \ add\text{-right-cancel} \ add\text{-closed} \ zero)$
 with vs show $x = -z$ by $(simp \ add: \ add\text{-minus-eq-minus})$
 next
 assume $eq: x = -z$
 then have $x + (y + z) = -z + (y + z)$ by *simp*
 also have $\dots = y + (-z + z)$
 by $(rule \ add\text{-left-commute}) \ (simp\text{-all} \ add: \ vs)$
 also from vs have $\dots = y$ by *simp*

```

    finally show  $x + (y + z) = y$  .
  }
qed

end

```

4 Subspaces

```

theory Subspace
imports VectorSpace
begin

```

4.1 Definition

A non-empty subset U of a vector space V is a *subspace* of V , iff U is closed under addition and scalar multiplication.

```

locale subspace =
  fixes  $U :: 'a::\{minus, plus, zero, uminus\}$  set and  $V$ 
  assumes non-empty [iff, intro]:  $U \neq \{\}$ 
    and subset [iff]:  $U \subseteq V$ 
    and add-closed [iff]:  $x \in U \implies y \in U \implies x + y \in U$ 
    and mult-closed [iff]:  $x \in U \implies a \cdot x \in U$ 

```

```

notation (symbols)
  subspace (infix  $\trianglelefteq$  50)

```

```

declare vectorspace.intro [intro?] subspace.intro [intro?]

```

```

lemma subspace-subset [elim]:  $U \trianglelefteq V \implies U \subseteq V$ 
  by (rule subspace.subset)

```

```

lemma (in subspace) subsetD [iff]:  $x \in U \implies x \in V$ 
  using subset by blast

```

```

lemma subspaceD [elim]:  $U \trianglelefteq V \implies x \in U \implies x \in V$ 
  by (rule subspace.subsetD)

```

```

lemma rev-subspaceD [elim?]:  $x \in U \implies U \trianglelefteq V \implies x \in V$ 
  by (rule subspace.subsetD)

```

```

lemma (in subspace) diff-closed [iff]:
  assumes vectorspace  $V$ 
  assumes  $x: x \in U$  and  $y: y \in U$ 
  shows  $x - y \in U$ 
proof -
  interpret vectorspace  $V$  by fact
  from  $x\ y$  show ?thesis by (simp add: diff-eq1 negate-eq1)
qed

```

Similar as for linear spaces, the existence of the zero element in every subspace follows from the non-emptiness of the carrier set and by vector space laws.

```

lemma (in subspace) zero [intro]:
  assumes vectorspace V
  shows  $0 \in U$ 
proof -
  interpret V: vectorspace V by fact
  have  $U \neq \{\}$  by (rule non-empty)
  then obtain x where  $x: x \in U$  by blast
  then have  $x \in V$  .. then have  $0 = x - x$  by simp
  also from (vectorspace V) x x have  $\dots \in U$  by (rule diff-closed)
  finally show ?thesis .
qed

```

```

lemma (in subspace) neg-closed [iff]:
  assumes vectorspace V
  assumes  $x: x \in U$ 
  shows  $-x \in U$ 
proof -
  interpret vectorspace V by fact
  from x show ?thesis by (simp add: negate-eq1)
qed

```

Further derived laws: every subspace is a vector space.

```

lemma (in subspace) vectorspace [iff]:
  assumes vectorspace V
  shows vectorspace U
proof -
  interpret vectorspace V by fact
  show ?thesis
proof
  show  $U \neq \{\}$  ..
  fix x y z assume  $x: x \in U$  and  $y: y \in U$  and  $z: z \in U$ 
  fix a b :: real
  from x y show  $x + y \in U$  by simp
  from x show  $a \cdot x \in U$  by simp
  from x y z show  $(x + y) + z = x + (y + z)$  by (simp add: add-ac)
  from x y show  $x + y = y + x$  by (simp add: add-ac)
  from x show  $x - x = 0$  by simp
  from x show  $0 + x = x$  by simp
  from x y show  $a \cdot (x + y) = a \cdot x + a \cdot y$  by (simp add: distrib)
  from x show  $(a + b) \cdot x = a \cdot x + b \cdot x$  by (simp add: distrib)
  from x show  $(a * b) \cdot x = a \cdot b \cdot x$  by (simp add: mult-assoc)
  from x show  $1 \cdot x = x$  by simp
  from x show  $-x = -1 \cdot x$  by (simp add: negate-eq1)
  from x y show  $x - y = x + -y$  by (simp add: diff-eq1)
qed
qed

```

The subspace relation is reflexive.

```

lemma (in vectorspace) subspace-refl [intro]:  $V \trianglelefteq V$ 
proof
  show  $V \neq \{\}$  ..
  show  $V \subseteq V$  ..
  fix x y assume  $x: x \in V$  and  $y: y \in V$ 

```

```

fix a :: real
from x y show x + y ∈ V by simp
from x show a · x ∈ V by simp
qed

```

The subspace relation is transitive.

```

lemma (in vectorspace) subspace-trans [trans]:
  U ≤ V ⇒ V ≤ W ⇒ U ≤ W
proof
  assume uv: U ≤ V and vw: V ≤ W
  from uv show U ≠ {} by (rule subspace.non-empty)
  show U ⊆ W
  proof -
    from uv have U ⊆ V by (rule subspace.subset)
    also from vw have V ⊆ W by (rule subspace.subset)
    finally show ?thesis .
  qed
  fix x y assume x: x ∈ U and y: y ∈ U
  from uv and x y show x + y ∈ U by (rule subspace.add-closed)
  from uv and x show ∧ a. a · x ∈ U by (rule subspace.mult-closed)
qed

```

4.2 Linear closure

The *linear closure* of a vector x is the set of all scalar multiples of x .

```

definition
  lin :: ('a::{minus, plus, zero}) ⇒ 'a set where
  lin x = {a · x | a. True}

```

```

lemma linI [intro]: y = a · x ⇒ y ∈ lin x
  unfolding lin-def by blast

```

```

lemma linI' [iff]: a · x ∈ lin x
  unfolding lin-def by blast

```

```

lemma linE [elim]: x ∈ lin v ⇒ (∧ a::real. x = a · v ⇒ C) ⇒ C
  unfolding lin-def by blast

```

Every vector is contained in its linear closure.

```

lemma (in vectorspace) x-lin-x [iff]: x ∈ V ⇒ x ∈ lin x
proof -
  assume x ∈ V
  then have x = 1 · x by simp
  also have ... ∈ lin x ..
  finally show ?thesis .
qed

```

```

lemma (in vectorspace) 0-lin-x [iff]: x ∈ V ⇒ 0 ∈ lin x
proof
  assume x ∈ V
  then show 0 = 0 · x by simp
qed

```

Any linear closure is a subspace.

```

lemma (in vectorspace) lin-subspace [intro]:
   $x \in V \implies \text{lin } x \subseteq V$ 
proof
  assume  $x: x \in V$ 
  then show  $\text{lin } x \neq \{\}$  by (auto simp add: x-lin-x)
  show  $\text{lin } x \subseteq V$ 
  proof
    fix  $x'$  assume  $x' \in \text{lin } x$ 
    then obtain  $a$  where  $x' = a \cdot x$  ..
    with  $x$  show  $x' \in V$  by simp
  qed
  fix  $x' x''$  assume  $x': x' \in \text{lin } x$  and  $x'': x'' \in \text{lin } x$ 
  show  $x' + x'' \in \text{lin } x$ 
  proof -
    from  $x'$  obtain  $a'$  where  $x' = a' \cdot x$  ..
    moreover from  $x''$  obtain  $a''$  where  $x'' = a'' \cdot x$  ..
    ultimately have  $x' + x'' = (a' + a'') \cdot x$ 
    using  $x$  by (simp add: distrib)
    also have  $\dots \in \text{lin } x$  ..
    finally show ?thesis .
  qed
  fix  $a :: \text{real}$ 
  show  $a \cdot x' \in \text{lin } x$ 
  proof -
    from  $x'$  obtain  $a'$  where  $x' = a' \cdot x$  ..
    with  $x$  have  $a \cdot x' = (a * a') \cdot x$  by (simp add: mult-assoc)
    also have  $\dots \in \text{lin } x$  ..
    finally show ?thesis .
  qed
qed

```

Any linear closure is a vector space.

```

lemma (in vectorspace) lin-vectorspace [intro]:
  assumes  $x \in V$ 
  shows vectorspace ( $\text{lin } x$ )
proof -
  from  $\langle x \in V \rangle$  have subspace ( $\text{lin } x$ )  $V$ 
  by (rule lin-subspace)
  from this and vectorspace-axioms show ?thesis
  by (rule subspace.vectorspace)
qed

```

4.3 Sum of two vectorspaces

The *sum* of two vectorspaces U and V is the set of all sums of elements from U and V .

```

instantiation fun :: (type, type) plus
begin

```

definition

```

  sum-def: plus-fun  $U V = \{u + v \mid u \in U \wedge v \in V\}$ 

```


instance ..

end

lemma *sumE* [*elim*]:

$x \in U + V \implies (\bigwedge u v. x = u + v \implies u \in U \implies v \in V \implies C) \implies C$

unfolding *sum-def* **by** *blast*

lemma *sumI* [*intro*]:

$u \in U \implies v \in V \implies x = u + v \implies x \in U + V$

unfolding *sum-def* **by** *blast*

lemma *sumI'* [*intro*]:

$u \in U \implies v \in V \implies u + v \in U + V$

unfolding *sum-def* **by** *blast*

U is a subspace of $U + V$.

lemma *subspace-sum1* [*iff*]:

assumes *vectorpace* U *vectorpace* V

shows $U \trianglelefteq U + V$

proof –

interpret *vectorpace* U **by** *fact*

interpret *vectorpace* V **by** *fact*

show *?thesis*

proof

show $U \neq \{\}$..

show $U \subseteq U + V$

proof

fix x **assume** $x: x \in U$

moreover **have** $0 \in V$..

ultimately **have** $x + 0 \in U + V$..

with x **show** $x \in U + V$ **by** *simp*

qed

fix $x y$ **assume** $x: x \in U$ **and** $y \in U$

then **show** $x + y \in U$ **by** *simp*

from x **show** $\bigwedge a. a \cdot x \in U$ **by** *simp*

qed

qed

The sum of two subspaces is again a subspace.

lemma *sum-subspace* [*intro?*]:

assumes *subspace* U E *vectorpace* E *subspace* V E

shows $U + V \trianglelefteq E$

proof –

interpret *subspace* U E **by** *fact*

interpret *vectorpace* E **by** *fact*

interpret *subspace* V E **by** *fact*

show *?thesis*

proof

have $0 \in U + V$

proof

show $0 \in U$ **using** $\langle \text{vectorpace } E \rangle$..

show $0 \in V$ **using** $\langle \text{vectorpace } E \rangle$..

show $(0::'a) = 0 + 0$ **by** *simp*

```

qed
then show  $U + V \neq \{\}$  by blast
show  $U + V \subseteq E$ 
proof
  fix x assume  $x \in U + V$ 
  then obtain u v where  $x = u + v$  and
     $u \in U$  and  $v \in V$  ..
  then show  $x \in E$  by simp
qed
fix x y assume  $x: x \in U + V$  and  $y: y \in U + V$ 
show  $x + y \in U + V$ 
proof -
  from x obtain ux vx where  $x = ux + vx$  and  $ux \in U$  and  $vx \in V$  ..
  moreover
  from y obtain uy vy where  $y = uy + vy$  and  $uy \in U$  and  $vy \in V$  ..
  ultimately
  have  $ux + uy \in U$ 
    and  $vx + vy \in V$ 
    and  $x + y = (ux + uy) + (vx + vy)$ 
    using x y by (simp-all add: add-ac)
  then show ?thesis ..
qed
fix a show  $a \cdot x \in U + V$ 
proof -
  from x obtain u v where  $x = u + v$  and  $u \in U$  and  $v \in V$  ..
  then have  $a \cdot u \in U$  and  $a \cdot v \in V$ 
    and  $a \cdot x = (a \cdot u) + (a \cdot v)$  by (simp-all add: distrib)
  then show ?thesis ..
qed
qed
qed

```

The sum of two subspaces is a vectorspace.

lemma *sum-vs* [intro?]:

$U \trianglelefteq E \implies V \trianglelefteq E \implies \text{vectorspace } E \implies \text{vectorspace } (U + V)$
 by (rule subspace.vectorspace) (rule sum-subspace)

4.4 Direct sums

The sum of U and V is called *direct*, iff the zero element is the only common element of U and V . For every element x of the direct sum of U and V the decomposition in $x = u + v$ with $u \in U$ and $v \in V$ is unique.

lemma *decomp*:

```

assumes vectorspace E subspace U E subspace V E
assumes direct:  $U \cap V = \{0\}$ 
  and u1:  $u1 \in U$  and u2:  $u2 \in U$ 
  and v1:  $v1 \in V$  and v2:  $v2 \in V$ 
  and sum:  $u1 + v1 = u2 + v2$ 
shows  $u1 = u2 \wedge v1 = v2$ 
proof -
  interpret vectorspace E by fact
  interpret subspace U E by fact
  interpret subspace V E by fact

```

```

show ?thesis
proof
  have  $U$ : vectorspace  $U$ 
    using  $\langle \text{subspace } U \ E \rangle \langle \text{vectorspace } E \rangle$  by (rule subspace.vectorspace)
  have  $V$ : vectorspace  $V$ 
    using  $\langle \text{subspace } V \ E \rangle \langle \text{vectorspace } E \rangle$  by (rule subspace.vectorspace)
  from  $u1 \ u2 \ v1 \ v2$  and sum have  $eq$ :  $u1 - u2 = v2 - v1$ 
    by (simp add: add-diff-swap)
  from  $u1 \ u2$  have  $u$ :  $u1 - u2 \in U$ 
    by (rule vectorspace.diff-closed [OF  $U$ ])
  with  $eq$  have  $v'$ :  $v2 - v1 \in U$  by (simp only:)
  from  $v2 \ v1$  have  $v$ :  $v2 - v1 \in V$ 
    by (rule vectorspace.diff-closed [OF  $V$ ])
  with  $eq$  have  $u'$ :  $u1 - u2 \in V$  by (simp only:)

  show  $u1 = u2$ 
  proof (rule add-minus-eq)
    from  $u1$  show  $u1 \in E$  ..
    from  $u2$  show  $u2 \in E$  ..
    from  $u \ u'$  and direct show  $u1 - u2 = 0$  by blast
  qed
  show  $v1 = v2$ 
  proof (rule add-minus-eq [symmetric])
    from  $v1$  show  $v1 \in E$  ..
    from  $v2$  show  $v2 \in E$  ..
    from  $v \ v'$  and direct show  $v2 - v1 = 0$  by blast
  qed
qed
qed

```

An application of the previous lemma will be used in the proof of the Hahn-Banach Theorem (see page 42): for any element $y + a \cdot x_0$ of the direct sum of a vectorspace H and the linear closure of x_0 the components $y \in H$ and a are uniquely determined.

```

lemma decomp-H':
  assumes vectorspace  $E$  subspace  $H \ E$ 
  assumes  $y1$ :  $y1 \in H$  and  $y2$ :  $y2 \in H$ 
    and  $x'$ :  $x' \notin H$   $x' \in E$   $x' \neq 0$ 
    and  $eq$ :  $y1 + a1 \cdot x' = y2 + a2 \cdot x'$ 
  shows  $y1 = y2 \wedge a1 = a2$ 
proof -
  interpret vectorspace  $E$  by fact
  interpret subspace  $H \ E$  by fact
  show ?thesis
  proof
    have  $c$ :  $y1 = y2 \wedge a1 \cdot x' = a2 \cdot x'$ 
    proof (rule decomp)
      show  $a1 \cdot x' \in \text{lin } x'$  ..
      show  $a2 \cdot x' \in \text{lin } x'$  ..
      show  $H \cap \text{lin } x' = \{0\}$ 
    proof
      show  $H \cap \text{lin } x' \subseteq \{0\}$ 
    proof
      fix  $x$  assume  $x$ :  $x \in H \cap \text{lin } x'$ 

```

```

then obtain  $a$  where  $xx'$ :  $x = a \cdot x'$ 
  by blast
have  $x = 0$ 
proof cases
  assume  $a = 0$ 
  with  $xx'$  and  $x'$  show ?thesis by simp
next
  assume  $a: a \neq 0$ 
  from  $x$  have  $x \in H$  ..
  with  $xx'$  have  $\text{inverse } a \cdot a \cdot x' \in H$  by simp
  with  $a$  and  $x'$  have  $x' \in H$  by (simp add: mult-assoc2)
  with  $\langle x' \notin H \rangle$  show ?thesis by contradiction
qed
then show  $x \in \{0\}$  ..
qed
show  $\{0\} \subseteq H \cap \text{lin } x'$ 
proof -
  have  $0 \in H$  using  $\langle \text{vectorspace } E \rangle$  ..
  moreover have  $0 \in \text{lin } x'$  using  $\langle x' \in E \rangle$  ..
  ultimately show ?thesis by blast
qed
qed
show  $\text{lin } x' \trianglelefteq E$  using  $\langle x' \in E \rangle$  ..
qed (rule  $\langle \text{vectorspace } E \rangle$ , rule  $\langle \text{subspace } H E \rangle$ , rule  $y1$ , rule  $y2$ , rule  $eq$ )
then show  $y1 = y2$  ..
from  $c$  have  $a1 \cdot x' = a2 \cdot x'$  ..
with  $x'$  show  $a1 = a2$  by (simp add: mult-right-cancel)
qed
qed

```

Since for any element $y + a \cdot x'$ of the direct sum of a vectorspace H and the linear closure of x' the components $y \in H$ and a are unique, it follows from $y \in H$ that $a = 0$.

lemma *decomp- $H'-H$:*

```

assumes  $\text{vectorspace } E \text{ subspace } H E$ 
assumes  $t: t \in H$ 
  and  $x': x' \notin H \ x' \in E \ x' \neq 0$ 
shows  $(\text{SOME } (y, a). t = y + a \cdot x' \wedge y \in H) = (t, 0)$ 
proof -
  interpret  $\text{vectorspace } E$  by fact
  interpret  $\text{subspace } H E$  by fact
  show ?thesis
proof (rule, simp-all only: split-paired-all split-conv)
  from  $t \ x'$  show  $t = t + 0 \cdot x' \wedge t \in H$  by simp
  fix  $y$  and  $a$  assume  $ya: t = y + a \cdot x' \wedge y \in H$ 
  have  $y = t \wedge a = 0$ 
  proof (rule decomp- $H'$ )
    from  $ya \ x'$  show  $y + a \cdot x' = t + 0 \cdot x'$  by simp
    from  $ya$  show  $y \in H$  ..
  qed (rule  $\langle \text{vectorspace } E \rangle$ , rule  $\langle \text{subspace } H E \rangle$ , rule  $t$ , (rule  $x'$ )+)
  with  $t \ x'$  show  $(y, a) = (y + a \cdot x', 0)$  by simp
qed
qed

```

The components $y \in H$ and a in $y + a \cdot x'$ are unique, so the function h' defined by $h'(y + a \cdot x') = h y + a \cdot \xi$ is definite.

```

lemma  $h'$ -definite:
  fixes  $H$ 
  assumes  $h'$ -def:
     $h' \equiv (\lambda x. \text{let } (y, a) = \text{SOME } (y, a). (x = y + a \cdot x' \wedge y \in H)$ 
       $\text{in } (h y) + a * xi)$ 
    and  $x: x = y + a \cdot x'$ 
  assumes vectorspace  $E$  subspace  $H E$ 
  assumes  $y: y \in H$ 
    and  $x': x' \notin H \ x' \in E \ x' \neq 0$ 
  shows  $h' x = h y + a * xi$ 
proof –
  interpret vectorspace  $E$  by fact
  interpret subspace  $H E$  by fact
  from  $x y x'$  have  $x \in H + \text{lin } x'$  by auto
  have  $\exists! p. (\lambda(y, a). x = y + a \cdot x' \wedge y \in H) p$  (is  $\exists! p. ?P p$  )
  proof (rule ex-ex1I)
    from  $x y$  show  $\exists p. ?P p$  by blast
    fix  $p q$  assume  $p: ?P p$  and  $q: ?P q$ 
    show  $p = q$ 
    proof –
      from  $p$  have  $xp: x = \text{fst } p + \text{snd } p \cdot x' \wedge \text{fst } p \in H$ 
      by (cases  $p$ ) simp
      from  $q$  have  $xq: x = \text{fst } q + \text{snd } q \cdot x' \wedge \text{fst } q \in H$ 
      by (cases  $q$ ) simp
      have  $\text{fst } p = \text{fst } q \wedge \text{snd } p = \text{snd } q$ 
      proof (rule decomp-H')
        from  $xp$  show  $\text{fst } p \in H$  ..
        from  $xq$  show  $\text{fst } q \in H$  ..
        from  $xp$  and  $xq$  show  $\text{fst } p + \text{snd } p \cdot x' = \text{fst } q + \text{snd } q \cdot x'$ 
        by simp
      qed (rule  $\langle \text{vectorspace } E \rangle$ , rule  $\langle \text{subspace } H E \rangle$ , (rule  $x'$ ) $+$ )
      then show  $?thesis$  by (cases  $p$ , cases  $q$ ) simp
    qed
  qed
  then have  $eq: (\text{SOME } (y, a). x = y + a \cdot x' \wedge y \in H) = (y, a)$ 
    by (rule some1-equality) (simp add:  $x y$ )
  with  $h'$ -def show  $h' x = h y + a * xi$  by (simp add: Let-def)
qed

end

```

5 Normed vector spaces

```

theory NormedSpace
imports Subspace
begin

```

5.1 Quasinorms

A *seminorm* $\|\cdot\|$ is a function on a real vector space into the reals that has the following properties: it is positive definite, absolute homogenous and subadditive.

```

locale norm-syntax =
  fixes norm :: 'a  $\Rightarrow$  real    ( $\|\cdot\|$ )

locale seminorm = var-V + norm-syntax +
  constrains V :: 'a::{minus, plus, zero, uminus} set
  assumes ge-zero [iff?]:  $x \in V \implies 0 \leq \|x\|$ 
    and abs-homogenous [iff?]:  $x \in V \implies \|a \cdot x\| = |a| * \|x\|$ 
    and subadditive [iff?]:  $x \in V \implies y \in V \implies \|x + y\| \leq \|x\| + \|y\|$ 

declare seminorm.intro [intro?]

lemma (in seminorm) diff-subadditive:
  assumes vectorspace V
  shows  $x \in V \implies y \in V \implies \|x - y\| \leq \|x\| + \|y\|$ 
proof -
  interpret vectorspace V by fact
  assume  $x: x \in V$  and  $y: y \in V$ 
  then have  $x - y = x + - 1 \cdot y$ 
    by (simp add: diff-eq2 negate-eq2a)
  also from  $x\ y$  have  $\|x - y\| \leq \|x\| + \|- 1 \cdot y\|$ 
    by (simp add: subadditive)
  also from  $y$  have  $\|- 1 \cdot y\| = |- 1| * \|y\|$ 
    by (rule abs-homogenous)
  also have  $\dots = \|y\|$  by simp
  finally show ?thesis .
qed

```

```

lemma (in seminorm) minus:
  assumes vectorspace V
  shows  $x \in V \implies \|- x\| = \|x\|$ 
proof -
  interpret vectorspace V by fact
  assume  $x: x \in V$ 
  then have  $- x = - 1 \cdot x$  by (simp only: negate-eq1)
  also from  $x$  have  $\|- x\| = |- 1| * \|x\|$ 
    by (rule abs-homogenous)
  also have  $\dots = \|x\|$  by simp
  finally show ?thesis .
qed

```

5.2 Norms

A *norm* $\|\cdot\|$ is a seminorm that maps only the 0 vector to 0.

```

locale norm = seminorm +
  assumes zero-iff [iff]:  $x \in V \implies (\|x\| = 0) = (x = 0)$ 

```

5.3 Normed vector spaces

A vector space together with a norm is called a *normed space*.

locale *normed-vectorspace* = *vectorspace* + *norm*

declare *normed-vectorspace.intro* [*intro?*]

lemma (**in** *normed-vectorspace*) *gt-zero* [*intro?*]:

$x \in V \implies x \neq 0 \implies 0 < \|x\|$

proof –

assume $x: x \in V$ **and** *neg*: $x \neq 0$

from x **have** $0 \leq \|x\|$ **..**

also have [*symmetric*]: $\dots \neq 0$

proof

assume $\|x\| = 0$

with x **have** $x = 0$ **by** *simp*

with *neg* **show** *False* **by** *contradiction*

qed

finally show *?thesis* .

qed

Any subspace of a normed vector space is again a normed vectorspace.

lemma *subspace-normed-vs* [*intro?*]:

fixes $F E$ *norm*

assumes *subspace* $F E$ *normed-vectorspace* E *norm*

shows *normed-vectorspace* F *norm*

proof –

interpret *subspace* $F E$ **by** *fact*

interpret *normed-vectorspace* E *norm* **by** *fact*

show *?thesis*

proof

show *vectorspace* F **by** (*rule vectorspace*) *unfold-locales*

next

have *NormedSpace.norm* E *norm* **..**

with *subset* **show** *NormedSpace.norm* F *norm*

by (*simp add: norm-def seminorm-def norm-axioms-def*)

qed

qed

end

6 Linearforms

theory *Linearform*

imports *VectorSpace*

begin

A *linear form* is a function on a vector space into the reals that is additive and multiplicative.

locale *linearform* =

fixes $V :: 'a::\{\text{minus}, \text{plus}, \text{zero}, \text{uminus}\}$ *set* **and** f

```

assumes add [iff]:  $x \in V \implies y \in V \implies f(x + y) = f x + f y$ 
and mult [iff]:  $x \in V \implies f(a \cdot x) = a * f x$ 

declare linearform.intro [intro?]

lemma (in linearform) neg [iff]:
  assumes vectorspace V
  shows  $x \in V \implies f(-x) = -f x$ 
proof -
  interpret vectorspace V by fact
  assume x:  $x \in V$ 
  then have  $f(-x) = f((-1) \cdot x)$  by (simp add: negate-eq1)
  also from x have  $\dots = (-1) * (f x)$  by (rule mult)
  also from x have  $\dots = -(f x)$  by simp
  finally show ?thesis .
qed

lemma (in linearform) diff [iff]:
  assumes vectorspace V
  shows  $x \in V \implies y \in V \implies f(x - y) = f x - f y$ 
proof -
  interpret vectorspace V by fact
  assume x:  $x \in V$  and y:  $y \in V$ 
  then have  $x - y = x + -y$  by (rule diff-eq1)
  also have  $f \dots = f x + f(-y)$  by (rule add) (simp-all add: x y)
  also have  $f(-y) = -f y$  using  $\langle \text{vectorspace } V \rangle y$  by (rule neg)
  finally show ?thesis by simp
qed

Every linear form yields 0 for the 0 vector.

lemma (in linearform) zero [iff]:
  assumes vectorspace V
  shows  $f 0 = 0$ 
proof -
  interpret vectorspace V by fact
  have  $f 0 = f(0 - 0)$  by simp
  also have  $\dots = f 0 - f 0$  using  $\langle \text{vectorspace } V \rangle$  by (rule diff) simp-all
  also have  $\dots = 0$  by simp
  finally show ?thesis .
qed

end

```

7 An order on functions

```

theory FunctionOrder
imports Subspace Linearform
begin

```

7.1 The graph of a function

We define the *graph* of a (real) function f with domain F as the set

$$\{(x, f x). x \in F\}$$

So we are modeling partial functions by specifying the domain and the mapping function. We use the term “function” also for its graph.

types 'a graph = ('a × real) set

definition

graph :: 'a set ⇒ ('a ⇒ real) ⇒ 'a graph **where**
graph F f = {(x, f x) | x. x ∈ F}

lemma graphI [intro]: x ∈ F ⇒ (x, f x) ∈ graph F f
unfolding graph-def **by** blast

lemma graphI2 [intro?]: x ∈ F ⇒ ∃ t ∈ graph F f. t = (x, f x)
unfolding graph-def **by** blast

lemma graphE [elim?]:
(x, y) ∈ graph F f ⇒ (x ∈ F ⇒ y = f x ⇒ C) ⇒ C
unfolding graph-def **by** blast

7.2 Functions ordered by domain extension

A function h' is an extension of h , iff the graph of h is a subset of the graph of h' .

lemma graph-extI:
($\bigwedge x. x \in H \Rightarrow h x = h' x$) ⇒ $H \subseteq H'$
⇒ graph H h ⊆ graph H' h'
unfolding graph-def **by** blast

lemma graph-extD1 [dest?]:
graph H h ⊆ graph H' h' ⇒ x ∈ H ⇒ h x = h' x
unfolding graph-def **by** blast

lemma graph-extD2 [dest?]:
graph H h ⊆ graph H' h' ⇒ H ⊆ H'
unfolding graph-def **by** blast

7.3 Domain and function of a graph

The inverse functions to *graph* are *domain* and *funct*.

definition

domain :: 'a graph ⇒ 'a set **where**
domain g = {x. ∃ y. (x, y) ∈ g}

definition

funct :: 'a graph ⇒ ('a ⇒ real) **where**
funct g = (λx. (SOME y. (x, y) ∈ g))

The following lemma states that g is the graph of a function if the relation induced by g is unique.

lemma graph-domain-funct:
assumes uniq: $\bigwedge x y z. (x, y) \in g \Rightarrow (x, z) \in g \Rightarrow z = y$

```

shows graph (domain g) (funct g) = g
unfolding domain-def funct-def graph-def
proof auto
  fix a b assume g: (a, b) ∈ g
  from g show (a, SOME y. (a, y) ∈ g) ∈ g by (rule someI2)
  from g show ∃ y. (a, y) ∈ g ..
  from g show b = (SOME y. (a, y) ∈ g)
  proof (rule some-equality [symmetric])
    fix y assume (a, y) ∈ g
    with g show y = b by (rule uniq)
  qed
qed

```

7.4 Norm-preserving extensions of a function

Given a linear form f on the space F and a seminorm p on E . The set of all linear extensions of f , to superspaces H of F , which are bounded by p , is defined as follows.

definition

```

norm-pres-extensions ::
  'a::{plus, minus, uminus, zero} set ⇒ ('a ⇒ real) ⇒ 'a set ⇒ ('a ⇒ real)
  ⇒ 'a graph set where
norm-pres-extensions E p F f
  = {g. ∃ H h. g = graph H h
    ∧ linearform H h
    ∧ H ⊆ E
    ∧ F ⊆ H
    ∧ graph F f ⊆ graph H h
    ∧ (∀ x ∈ H. h x ≤ p x)}

```

lemma norm-pres-extensionE [elim]:

```

g ∈ norm-pres-extensions E p F f
⇒ (⋀ H h. g = graph H h ⇒ linearform H h
  ⇒ H ⊆ E ⇒ F ⊆ H ⇒ graph F f ⊆ graph H h
  ⇒ ∀ x ∈ H. h x ≤ p x ⇒ C) ⇒ C
unfolding norm-pres-extensions-def by blast

```

lemma norm-pres-extensionI2 [intro]:

```

linearform H h ⇒ H ⊆ E ⇒ F ⊆ H
⇒ graph F f ⊆ graph H h ⇒ ∀ x ∈ H. h x ≤ p x
⇒ graph H h ∈ norm-pres-extensions E p F f
unfolding norm-pres-extensions-def by blast

```

lemma norm-pres-extensionI:

```

∃ H h. g = graph H h
  ∧ linearform H h
  ∧ H ⊆ E
  ∧ F ⊆ H
  ∧ graph F f ⊆ graph H h
  ∧ (∀ x ∈ H. h x ≤ p x) ⇒ g ∈ norm-pres-extensions E p F f
unfolding norm-pres-extensions-def by blast

```

end

8 The norm of a function

```
theory FunctionNorm
imports NormedSpace FunctionOrder
begin
```

8.1 Continuous linear forms

A linear form f on a normed vector space $(V, \|\cdot\|)$ is *continuous*, iff it is bounded, i.e.

$$\exists c \in \mathbb{R}. \forall x \in V. |f x| \leq c \cdot \|x\|$$

In our application no other functions than linear forms are considered, so we can define continuous linear forms as bounded linear forms:

```
locale continuous = var- $V$  + norm-syntax + linearform +
  assumes bounded:  $\exists c. \forall x \in V. |f x| \leq c * \|x\|$ 

declare continuous.intro [intro?] continuous-axioms.intro [intro?]

lemma continuousI [intro]:
  fixes norm ::  $- \Rightarrow \text{real}$  ( $\|\cdot\|$ )
  assumes linearform  $V f$ 
  assumes  $r$ :  $\bigwedge x. x \in V \Longrightarrow |f x| \leq c * \|x\|$ 
  shows continuous  $V$  norm  $f$ 
proof
  show linearform  $V f$  by fact
  from  $r$  have  $\exists c. \forall x \in V. |f x| \leq c * \|x\|$  by blast
  then show continuous-axioms  $V$  norm  $f$  ..
qed
```

8.2 The norm of a linear form

The least real number c for which holds

$$\forall x \in V. |f x| \leq c \cdot \|x\|$$

is called the *norm* of f .

For non-trivial vector spaces $V \neq \{0\}$ the norm can be defined as

$$\|f\| = \sup_{x \neq 0} |f x| / \|x\|$$

For the case $V = \{0\}$ the supremum would be taken from an empty set. Since \mathbb{R} is unbounded, there would be no supremum. To avoid this situation it must be guaranteed that there is an element in this set. This element must be ≥ 0 so that *fn-norm* has the norm properties. Furthermore it does not have to change the norm in all other cases, so it must be 0 , as all other elements are ≥ 0 .

Thus we define the set B where the supremum is taken from as follows:

$$\{0\} \cup \{|f x| / \|x\|. \ x \neq 0 \wedge x \in F\}$$

fn-norm is equal to the supremum of B , if the supremum exists (otherwise it is undefined).

```

locale fn-norm = norm-syntax +
  fixes  $B$  defines  $B \ V f \equiv \{0\} \cup \{|f x| / \|x\| \mid x. x \neq 0 \wedge x \in V\}$ 
  fixes fn-norm ( $\| \cdot \|$ ) --  $[0, 1000] \ 999$ 
  defines  $\|f\| - V \equiv \bigsqcup (B \ V f)$ 

```

```

locale normed-vectorspace-with-fn-norm = normed-vectorspace + fn-norm

```

```

lemma (in fn-norm) B-not-empty [intro]:  $0 \in B \ V f$ 
  by (simp add: B-def)

```

The following lemma states that every continuous linear form on a normed space $(V, \|\cdot\|)$ has a function norm.

```

lemma (in normed-vectorspace-with-fn-norm) fn-norm-works:
  assumes continuous V norm f
  shows  $\text{lub } (B \ V f) (\|f\| - V)$ 
proof –
  interpret continuous V norm f by fact

```

The existence of the supremum is shown using the completeness of the reals. Completeness means, that every non-empty bounded set of reals has a supremum.

```

  have  $\exists a. \text{lub } (B \ V f) \ a$ 
  proof (rule real-complete)

```

First we have to show that B is non-empty:

```

  have  $0 \in B \ V f \ ..$ 
  then show  $\exists x. x \in B \ V f \ ..$ 

```

Then we have to show that B is bounded:

```

  show  $\exists c. \forall y \in B \ V f. y \leq c$ 
  proof –

```

We know that f is bounded by some value c .

```

  from bounded obtain  $c$  where  $c: \forall x \in V. |f x| \leq c * \|x\| \ ..$ 

```

To prove the thesis, we have to show that there is some b , such that $y \leq b$ for all $y \in B$. Due to the definition of B there are two cases.

```

  def  $b \equiv \max \ c \ 0$ 
  have  $\forall y \in B \ V f. y \leq b$ 
  proof
    fix  $y$  assume  $y: y \in B \ V f$ 
    show  $y \leq b$ 
    proof cases
      assume  $y = 0$ 
      then show ?thesis unfolding b-def by arith
    next

```

The second case is $y = |f x| / \|x\|$ for some $x \in V$ with $x \neq 0$.

```

  assume  $y \neq 0$ 

```

```

with y obtain x where y-rep: y = |f x| * inverse ||x||
and x: x ∈ V and neg: x ≠ 0
by (auto simp add: B-def real-divide-def)
from x neg have gt: 0 < ||x|| ..

```

The thesis follows by a short calculation using the fact that f is bounded.

```

note y-rep
also have |f x| * inverse ||x|| ≤ (c * ||x||) * inverse ||x||
proof (rule mult-right-mono)
  from c x show |f x| ≤ c * ||x|| ..
  from gt have 0 < inverse ||x||
  by (rule positive-imp-inverse-positive)
  then show 0 ≤ inverse ||x|| by (rule order-less-imp-le)
qed
also have ... = c * (||x|| * inverse ||x||)
  by (rule real-mult-assoc)
also
from gt have ||x|| ≠ 0 by simp
then have ||x|| * inverse ||x|| = 1 by simp
also have c * 1 ≤ b by (simp add: b-def le-maxI1)
finally show y ≤ b .
qed
qed
then show ?thesis ..
qed
qed
then show ?thesis unfolding fn-norm-def by (rule the-lubI-ex)
qed

```

```

lemma (in normed-vectorspace-with-fn-norm) fn-norm-ub [iff?]:
  assumes continuous V norm f
  assumes b: b ∈ B V f
  shows b ≤ ||f||-V
proof -
  interpret continuous V norm f by fact
  have lub (B V f) (||f||-V)
    using ⟨continuous V norm f⟩ by (rule fn-norm-works)
  from this and b show ?thesis ..
qed

```

```

lemma (in normed-vectorspace-with-fn-norm) fn-norm-leastB:
  assumes continuous V norm f
  assumes b: ⋀b. b ∈ B V f ⟹ b ≤ y
  shows ||f||-V ≤ y
proof -
  interpret continuous V norm f by fact
  have lub (B V f) (||f||-V)
    using ⟨continuous V norm f⟩ by (rule fn-norm-works)
  from this and b show ?thesis ..
qed

```

The norm of a continuous function is always ≥ 0 .

```

lemma (in normed-vectorspace-with-fn-norm) fn-norm-ge-zero [iff]:
  assumes continuous V norm f

```

```

  shows  $0 \leq \|f\| - V$ 
proof -
  interpret continuous V norm f by fact

```

The function norm is defined as the supremum of B . So it is ≥ 0 if all elements in B are ≥ 0 , provided the supremum exists and B is not empty.

```

  have  $\text{lub } (B \ V \ f) \ (\|f\| - V)$ 
    using  $\langle \text{continuous } V \text{ norm } f \rangle$  by (rule fn-norm-works)
  moreover have  $0 \in B \ V \ f$  ..
  ultimately show ?thesis ..
qed

```

The fundamental property of function norms is:

$$|f \ x| \leq \|f\| \cdot \|x\|$$

lemma (in *normed-vectorspace-with-fn-norm*) *fn-norm-le-cong*:

```

  assumes continuous V norm f linearform V f
  assumes  $x: x \in V$ 
  shows  $|f \ x| \leq \|f\| - V * \|x\|$ 
proof -
  interpret continuous V norm f by fact
  interpret linearform V f by fact
  show ?thesis
proof cases
  assume  $x = 0$ 
  then have  $|f \ x| = |f \ 0|$  by simp
  also have  $f \ 0 = 0$  by rule unfold-locales
  also have  $|\dots| = 0$  by simp
  also have  $a: 0 \leq \|f\| - V$ 
    using  $\langle \text{continuous } V \text{ norm } f \rangle$  by (rule fn-norm-ge-zero)
  from  $x$  have  $0 \leq \text{norm } x$  ..
  with  $a$  have  $0 \leq \|f\| - V * \|x\|$  by (simp add: zero-le-mult-iff)
  finally show  $|f \ x| \leq \|f\| - V * \|x\|$  .
next
  assume  $x \neq 0$ 
  with  $x$  have neg:  $\|x\| \neq 0$  by simp
  then have  $|f \ x| = (|f \ x| * \text{inverse } \|x\|) * \|x\|$  by simp
  also have  $\dots \leq \|f\| - V * \|x\|$ 
  proof (rule mult-right-mono)
    from  $x$  show  $0 \leq \|x\|$  ..
    from  $x$  and neg have  $|f \ x| * \text{inverse } \|x\| \in B \ V \ f$ 
      by (auto simp add: B-def real-divide-def)
    with  $\langle \text{continuous } V \text{ norm } f \rangle$  show  $|f \ x| * \text{inverse } \|x\| \leq \|f\| - V$ 
      by (rule fn-norm-ub)
  qed
  finally show ?thesis .
qed
qed

```

The function norm is the least positive real number for which the following inequation holds:

$$|f \ x| \leq c \cdot \|x\|$$

```

lemma (in normed-vectorspace-with-fn-norm) fn-norm-least [intro?]:
  assumes continuous V norm f
  assumes ineq:  $\forall x \in V. |f x| \leq c * \|x\|$  and ge:  $0 \leq c$ 
  shows  $\|f\| - V \leq c$ 
proof -
  interpret continuous V norm f by fact
  show ?thesis
  proof (rule fn-norm-leastB [folded B-def fn-norm-def])
    fix b assume b:  $b \in B \ V f$ 
    show  $b \leq c$ 
    proof cases
      assume  $b = 0$ 
      with ge show ?thesis by simp
    next
      assume  $b \neq 0$ 
      with b obtain x where b-rep:  $b = |f x| * \text{inverse } \|x\|$ 
        and x-neq:  $x \neq 0$  and x:  $x \in V$ 
        by (auto simp add: B-def real-divide-def)
      note b-rep
      also have  $|f x| * \text{inverse } \|x\| \leq (c * \|x\|) * \text{inverse } \|x\|$ 
      proof (rule mult-right-mono)
        have  $0 < \|x\|$  using x-neq ..
        then show  $0 \leq \text{inverse } \|x\|$  by simp
        from ineq and x show  $|f x| \leq c * \|x\|$  ..
      qed
      also have  $\dots = c$ 
      proof -
        from x-neq and x have  $\|x\| \neq 0$  by simp
        then show ?thesis by simp
      qed
      finally show ?thesis .
    qed
  qed (insert (continuous V norm f), simp-all add: continuous-def)
qed

end

```

9 Zorn's Lemma

```

theory ZornLemma
imports Zorn
begin

```

Zorn's Lemmas states: if every linear ordered subset of an ordered set S has an upper bound in S , then there exists a maximal element in S . In our application, S is a set of sets ordered by set inclusion. Since the union of a chain of sets is an upper bound for all elements of the chain, the conditions of Zorn's lemma can be modified: if S is non-empty, it suffices to show that for every non-empty chain c in S the union of c also lies in S .

theorem *Zorn's-Lemma*:

```

assumes r:  $\bigwedge c. c \in \text{chain } S \implies \exists x. x \in c \implies \bigcup c \in S$ 
and aS:  $a \in S$ 

```

```

shows  $\exists y \in S. \forall z \in S. y \subseteq z \longrightarrow y = z$ 
proof (rule Zorn-Lemma2)
show  $\forall c \in \text{chain } S. \exists y \in S. \forall z \in c. z \subseteq y$ 
proof
  fix  $c$  assume  $c \in \text{chain } S$ 
  show  $\exists y \in S. \forall z \in c. z \subseteq y$ 
proof cases

```

If c is an empty chain, then every element in S is an upper bound of c .

```

assume  $c = \{\}$ 
with  $aS$  show ?thesis by fast

```

If c is non-empty, then $\bigcup c$ is an upper bound of c , lying in S .

```

next
assume  $c \neq \{\}$ 
show ?thesis
proof
  show  $\forall z \in c. z \subseteq \bigcup c$  by fast
  show  $\bigcup c \in S$ 
  proof (rule r)
    from  $\langle c \neq \{\} \rangle$  show  $\exists x. x \in c$  by fast
    show  $c \in \text{chain } S$  by fact
  qed
qed
qed
qed
qed
end

```


Part II

Lemmas for the Proof

10 The supremum w.r.t. the function order

```

theory HahnBanachSupLemmas
imports FunctionNorm ZornLemma
begin

```

This section contains some lemmas that will be used in the proof of the Hahn-Banach Theorem. In this section the following context is presumed. Let E be a real vector space with a seminorm p on E . F is a subspace of E and f a linear form on F . We consider a chain c of norm-preserving extensions of f , such that $\bigcup c = \text{graph } H \ h$. We will show some properties about the limit function h , i.e. the supremum of the chain c .

Let c be a chain of norm-preserving extensions of the function f and let $\text{graph } H \ h$ be the supremum of c . Every element in H is member of one of the elements of the chain.

```

lemmas [dest?] = chainD
lemmas chainE2 [elim?] = chainD2 [elim-format, standard]

```

lemma *some- $H' h'$:*

assumes M : $M = \text{norm-pres-extensions } E \ p \ F \ f$

and cM : $c \in \text{chain } M$

and u : $\text{graph } H \ h = \bigcup c$

and x : $x \in H$

shows $\exists H' h'. \text{graph } H' h' \in c$

$\wedge (x, h \ x) \in \text{graph } H' h'$

$\wedge \text{linearform } H' h' \wedge H' \trianglelefteq E$

$\wedge F \trianglelefteq H' \wedge \text{graph } F f \subseteq \text{graph } H' h'$

$\wedge (\forall x \in H'. h' \ x \leq p \ x)$

proof –

from x **have** $(x, h \ x) \in \text{graph } H \ h$..

also from u **have** $\dots = \bigcup c$.

finally obtain g **where** gc : $g \in c$ **and** gh : $(x, h \ x) \in g$ **by** *blast*

from cM **have** $c \subseteq M$..

with gc **have** $g \in M$..

also from M **have** $\dots = \text{norm-pres-extensions } E \ p \ F \ f$.

finally obtain H' **and** h' **where** g : $g = \text{graph } H' h'$

and $*$: $\text{linearform } H' h' \wedge H' \trianglelefteq E \wedge F \trianglelefteq H'$

$\text{graph } F f \subseteq \text{graph } H' h' \wedge \forall x \in H'. h' \ x \leq p \ x$..

from gc **and** g **have** $\text{graph } H' h' \in c$ **by** (*simp only*:)

moreover from gh **and** g **have** $(x, h \ x) \in \text{graph } H' h'$ **by** (*simp only*:)

ultimately show *?thesis* **using** $*$ **by** *blast*

qed

Let c be a chain of norm-preserving extensions of the function f and let $\text{graph } H \ h$ be the supremum of c . Every element in the domain H of the supremum

function is member of the domain H' of some function h' , such that h extends h' .

lemma *some- $H'h'$* :

assumes M : $M = \text{norm-pres-extensions } E \ p \ F \ f$
and cM : $c \in \text{chain } M$
and u : $\text{graph } H \ h = \bigcup c$
and x : $x \in H$
shows $\exists H' \ h'. x \in H' \wedge \text{graph } H' \ h' \subseteq \text{graph } H \ h$
 $\wedge \text{linearform } H' \ h' \wedge H' \trianglelefteq E \wedge F \trianglelefteq H'$
 $\wedge \text{graph } F \ f \subseteq \text{graph } H' \ h' \wedge (\forall x \in H'. h' \ x \leq p \ x)$

proof –

from $M \ cM \ u \ x$ **obtain** $H' \ h'$ **where**
 $x\text{-hx}$: $(x, h \ x) \in \text{graph } H' \ h'$
and c : $\text{graph } H' \ h' \in c$
and $*$: $\text{linearform } H' \ h' \ H' \trianglelefteq E \ F \trianglelefteq H'$
 $\text{graph } F \ f \subseteq \text{graph } H' \ h' \ \forall x \in H'. h' \ x \leq p \ x$
by (rule *some- $H'h'$ t* [elim-format]) **blast**
from $x\text{-hx}$ **have** $x \in H' \dots$
moreover from $cM \ u \ c$ **have** $\text{graph } H' \ h' \subseteq \text{graph } H \ h$
by (simp only: *chain-ball-Union-upper*)
ultimately show *?thesis* **using** $*$ **by** **blast**
qed

Any two elements x and y in the domain H of the supremum function h are both in the domain H' of some function h' , such that h extends h' .

lemma *some- $H'h'2$* :

assumes M : $M = \text{norm-pres-extensions } E \ p \ F \ f$
and cM : $c \in \text{chain } M$
and u : $\text{graph } H \ h = \bigcup c$
and x : $x \in H$
and y : $y \in H$
shows $\exists H' \ h'. x \in H' \wedge y \in H'$
 $\wedge \text{graph } H' \ h' \subseteq \text{graph } H \ h$
 $\wedge \text{linearform } H' \ h' \wedge H' \trianglelefteq E \wedge F \trianglelefteq H'$
 $\wedge \text{graph } F \ f \subseteq \text{graph } H' \ h' \wedge (\forall x \in H'. h' \ x \leq p \ x)$

proof –

y is in the domain H'' of some function h'' , such that h extends h'' .

from $M \ cM \ u$ **and** y **obtain** $H' \ h'$ **where**
 $y\text{-hy}$: $(y, h \ y) \in \text{graph } H' \ h'$
and c' : $\text{graph } H' \ h' \in c$
and $*$:
 $\text{linearform } H' \ h' \ H' \trianglelefteq E \ F \trianglelefteq H'$
 $\text{graph } F \ f \subseteq \text{graph } H' \ h' \ \forall x \in H'. h' \ x \leq p \ x$
by (rule *some- $H'h'$ t* [elim-format]) **blast**

x is in the domain H' of some function h' , such that h extends h' .

from $M \ cM \ u$ **and** x **obtain** $H'' \ h''$ **where**
 $x\text{-hx}$: $(x, h \ x) \in \text{graph } H'' \ h''$
and c'' : $\text{graph } H'' \ h'' \in c$
and $**$:
 $\text{linearform } H'' \ h'' \ H'' \trianglelefteq E \ F \trianglelefteq H''$

$\text{graph } F f \subseteq \text{graph } H'' h'' \quad \forall x \in H''. h'' x \leq p x$
by (*rule some- $H'h't$ [elim-format]*) *blast*

Since both h' and h'' are elements of the chain, h'' is an extension of h' or vice versa.
 Thus both x and y are contained in the greater one.

from $cM c'' c'$ **have** $\text{graph } H'' h'' \subseteq \text{graph } H' h' \vee \text{graph } H' h' \subseteq \text{graph } H'' h''$
 (*is ?case1 \vee ?case2*) ..
then show *?thesis*
proof
assume *?case1*
have $(x, h x) \in \text{graph } H'' h''$ **by** *fact*
also have $\dots \subseteq \text{graph } H' h'$ **by** *fact*
finally have $xh:(x, h x) \in \text{graph } H' h'$.
then have $x \in H'$..
moreover from $y-hy$ **have** $y \in H'$..
moreover from $cM u$ **and** c' **have** $\text{graph } H' h' \subseteq \text{graph } H h$
by (*simp only: chain-ball-Union-upper*)
ultimately show *?thesis* **using** $*$ **by** *blast*
next
assume *?case2*
from $x-hx$ **have** $x \in H''$..
moreover {
have $(y, h y) \in \text{graph } H' h'$ **by** (*rule y-hy*)
also have $\dots \subseteq \text{graph } H'' h''$ **by** *fact*
finally have $(y, h y) \in \text{graph } H'' h''$.
} **then have** $y \in H''$..
moreover from $cM u$ **and** c'' **have** $\text{graph } H'' h'' \subseteq \text{graph } H h$
by (*simp only: chain-ball-Union-upper*)
ultimately show *?thesis* **using** $**$ **by** *blast*
qed
qed

The relation induced by the graph of the supremum of a chain c is definite,
 i. e. t is the graph of a function.

lemma *sup-definite*:

assumes $M\text{-def}$: $M \equiv \text{norm-pres-extensions } E p F f$
and cM : $c \in \text{chain } M$
and xy : $(x, y) \in \bigcup c$
and xz : $(x, z) \in \bigcup c$
shows $z = y$

proof –

from cM **have** $c: c \subseteq M$..
from xy **obtain** $G1$ **where** $xy': (x, y) \in G1$ **and** $G1$: $G1 \in c$..
from xz **obtain** $G2$ **where** $xz': (x, z) \in G2$ **and** $G2$: $G2 \in c$..

from $G1 c$ **have** $G1 \in M$..
then obtain $H1 h1$ **where** $G1\text{-rep}$: $G1 = \text{graph } H1 h1$
unfolding $M\text{-def}$ **by** *blast*

from $G2 c$ **have** $G2 \in M$..
then obtain $H2 h2$ **where** $G2\text{-rep}$: $G2 = \text{graph } H2 h2$
unfolding $M\text{-def}$ **by** *blast*

G_1 is contained in G_2 or vice versa, since both G_1 and G_2 are members of c .

```

from  $cM$   $G1$   $G2$  have  $G1 \subseteq G2 \vee G2 \subseteq G1$  (is  $?case1 \vee ?case2$ ) ..
then show  $?thesis$ 
proof
  assume  $?case1$ 
  with  $xy'$   $G2$ -rep have  $(x, y) \in \text{graph } H2$   $h2$  by blast
  then have  $y = h2\ x$  ..
  also
  from  $xz'$   $G2$ -rep have  $(x, z) \in \text{graph } H2$   $h2$  by (simp only:)
  then have  $z = h2\ x$  ..
  finally show  $?thesis$  .
next
  assume  $?case2$ 
  with  $xz'$   $G1$ -rep have  $(x, z) \in \text{graph } H1$   $h1$  by blast
  then have  $z = h1\ x$  ..
  also
  from  $xy'$   $G1$ -rep have  $(x, y) \in \text{graph } H1$   $h1$  by (simp only:)
  then have  $y = h1\ x$  ..
  finally show  $?thesis$  ..
qed
qed

```

The limit function h is linear. Every element x in the domain of h is in the domain of a function h' in the chain of norm preserving extensions. Furthermore, h is an extension of h' so the function values of x are identical for h' and h . Finally, the function h' is linear by construction of M .

lemma *sup-lf*:

```

assumes  $M$ :  $M = \text{norm-pres-extensions } E\ p\ F\ f$ 
and  $cM$ :  $c \in \text{chain } M$ 
and  $u$ :  $\text{graph } H\ h = \bigcup c$ 
shows  $\text{linearform } H\ h$ 
proof
  fix  $x\ y$  assume  $x: x \in H$  and  $y: y \in H$ 
  with  $M\ cM\ u$  obtain  $H'\ h'$  where
     $x': x \in H'$  and  $y': y \in H'$ 
    and  $b$ :  $\text{graph } H'\ h' \subseteq \text{graph } H\ h$ 
    and  $\text{linearform}$ :  $\text{linearform } H'\ h'$ 
    and  $\text{subspace}$ :  $H' \trianglelefteq E$ 
    by (rule some- $H'h'2$  [elim-format]) blast

  show  $h\ (x + y) = h\ x + h\ y$ 
  proof –
    from  $\text{linearform } x'\ y'$  have  $h'\ (x + y) = h'\ x + h'\ y$ 
      by (rule linearform.add)
    also from  $b\ x'$  have  $h'\ x = h\ x$  ..
    also from  $b\ y'$  have  $h'\ y = h\ y$  ..
    also from  $\text{subspace } x'\ y'$  have  $x + y \in H'$ 
      by (rule subspace.add-closed)
    with  $b$  have  $h'\ (x + y) = h\ (x + y)$  ..
    finally show  $?thesis$  .
  qed
next
  fix  $x\ a$  assume  $x: x \in H$ 
  with  $M\ cM\ u$  obtain  $H'\ h'$  where

```

```

       $x': x \in H'$ 
    and  $b: \text{graph } H' h' \subseteq \text{graph } H h$ 
    and  $\text{linearform}: \text{linearform } H' h'$ 
    and  $\text{subspace}: H' \trianglelefteq E$ 
  by (rule some- $H'h'$  [elim-format]) blast

show  $h(a \cdot x) = a * h x$ 
proof -
  from  $\text{linearform } x'$  have  $h'(a \cdot x) = a * h' x$ 
  by (rule  $\text{linearform.mult}$ )
  also from  $b x'$  have  $h' x = h x$  ..
  also from  $\text{subspace } x'$  have  $a \cdot x \in H'$ 
  by (rule  $\text{subspace.mult-closed}$ )
  with  $b$  have  $h'(a \cdot x) = h(a \cdot x)$  ..
  finally show ?thesis .
qed
qed

```

The limit of a non-empty chain of norm preserving extensions of f is an extension of f , since every element of the chain is an extension of f and the supremum is an extension for every element of the chain.

```

lemma sup-ext:
  assumes  $\text{graph}: \text{graph } H h = \bigcup c$ 
  and  $M: M = \text{norm-pres-extensions } E p F f$ 
  and  $cM: c \in \text{chain } M$ 
  and  $ex: \exists x. x \in c$ 
  shows  $\text{graph } F f \subseteq \text{graph } H h$ 
proof -
  from  $ex$  obtain  $x$  where  $xc: x \in c$  ..
  from  $cM$  have  $c \subseteq M$  ..
  with  $xc$  have  $x \in M$  ..
  with  $M$  have  $x \in \text{norm-pres-extensions } E p F f$ 
  by (simp only:)
  then obtain  $G g$  where  $x = \text{graph } G g$  and  $\text{graph } F f \subseteq \text{graph } G g$  ..
  then have  $\text{graph } F f \subseteq x$  by (simp only:)
  also from  $xc$  have  $\dots \subseteq \bigcup c$  by blast
  also from  $\text{graph}$  have  $\dots = \text{graph } H h$  ..
  finally show ?thesis .
qed

```

The domain H of the limit function is a superspace of F , since F is a subset of H . The existence of the 0 element in F and the closure properties follow from the fact that F is a vector space.

```

lemma sup-supF:
  assumes  $\text{graph}: \text{graph } H h = \bigcup c$ 
  and  $M: M = \text{norm-pres-extensions } E p F f$ 
  and  $cM: c \in \text{chain } M$ 
  and  $ex: \exists x. x \in c$ 
  and  $FE: F \trianglelefteq E$ 
  shows  $F \trianglelefteq H$ 
proof
  from  $FE$  show  $F \neq \{\}$  by (rule  $\text{subspace.non-empty}$ )

```

```

from graph  $M$   $cM$   $ex$  have graph  $F$   $f \subseteq$  graph  $H$   $h$  by (rule sup-ext)
then show  $F \subseteq H$  ..
fix  $x$   $y$  assume  $x \in F$  and  $y \in F$ 
with  $FE$  show  $x + y \in F$  by (rule subspace.add-closed)
next
fix  $x$   $a$  assume  $x \in F$ 
with  $FE$  show  $a \cdot x \in F$  by (rule subspace.mult-closed)
qed

```

The domain H of the limit function is a subspace of E .

lemma *sup-subE*:

```

assumes graph: graph  $H$   $h = \bigcup c$ 
and  $M$ :  $M = \text{norm-pres-extensions } E$   $p$   $F$   $f$ 
and  $cM$ :  $c \in \text{chain } M$ 
and  $ex$ :  $\exists x. x \in c$ 
and  $FE$ :  $F \trianglelefteq E$ 
and  $E$ : vectorspace  $E$ 
shows  $H \trianglelefteq E$ 
proof
show  $H \neq \{\}$ 
proof –
from  $FE$   $E$  have  $0 \in F$  by (rule subspace.zero)
also from graph  $M$   $cM$   $ex$   $FE$  have  $F \trianglelefteq H$  by (rule sup-supF)
then have  $F \subseteq H$  ..
finally show ?thesis by blast
qed
show  $H \subseteq E$ 
proof
fix  $x$  assume  $x \in H$ 
with  $M$   $cM$  graph
obtain  $H'$   $h'$  where  $x: x \in H'$  and  $H'E: H' \trianglelefteq E$ 
by (rule some-H'h' [elim-format]) blast
from  $H'E$  have  $H' \subseteq E$  ..
with  $x$  show  $x \in E$  ..
qed
fix  $x$   $y$  assume  $x: x \in H$  and  $y: y \in H$ 
show  $x + y \in H$ 
proof –
from  $M$   $cM$  graph  $x$   $y$  obtain  $H'$   $h'$  where
 $x': x \in H'$  and  $y': y \in H'$  and  $H'E: H' \trianglelefteq E$ 
and graphs: graph  $H'$   $h' \subseteq$  graph  $H$   $h$ 
by (rule some-H'h'2 [elim-format]) blast
from  $H'E$   $x'$   $y'$  have  $x + y \in H'$ 
by (rule subspace.add-closed)
also from graphs have  $H' \subseteq H$  ..
finally show ?thesis .
qed
next
fix  $x$   $a$  assume  $x: x \in H$ 
show  $a \cdot x \in H$ 
proof –
from  $M$   $cM$  graph  $x$ 
obtain  $H'$   $h'$  where  $x': x \in H'$  and  $H'E: H' \trianglelefteq E$ 
and graphs: graph  $H'$   $h' \subseteq$  graph  $H$   $h$ 

```

```

    by (rule some-H'h' [elim-format]) blast
  from H'E x' have a · x ∈ H' by (rule subspace.mult-closed)
  also from graphs have H' ⊆ H ..
  finally show ?thesis .
qed
qed

```

The limit function is bounded by the norm p as well, since all elements in the chain are bounded by p .

```

lemma sup-norm-pres:
  assumes graph: graph H h = ⋃ c
    and M: M = norm-pres-extensions E p F f
    and cM: c ∈ chain M
  shows ∀ x ∈ H. h x ≤ p x
proof
  fix x assume x ∈ H
  with M cM graph obtain H' h' where x': x ∈ H'
    and graphs: graph H' h' ⊆ graph H h
    and a: ∀ x ∈ H'. h' x ≤ p x
    by (rule some-H'h' [elim-format]) blast
  from graphs x' have [symmetric]: h' x = h x ..
  also from a x' have h' x ≤ p x ..
  finally show h x ≤ p x .
qed

```

The following lemma is a property of linear forms on real vector spaces. It will be used for the lemma *abs-HahnBanach* (see page 51). For real vector spaces the following inequations are equivalent:

$$\forall x \in H. |h x| \leq p x \quad \text{and} \quad \forall x \in H. h x \leq p x$$

```

lemma abs-ineq-iff:
  assumes subspace H E and vectorspace E and seminorm E p
    and linearform H h
  shows (∀ x ∈ H. |h x| ≤ p x) = (∀ x ∈ H. h x ≤ p x) (is ?L = ?R)
proof
  interpret subspace H E by fact
  interpret vectorspace E by fact
  interpret seminorm E p by fact
  interpret linearform H h by fact
  have H: vectorspace H using ⟨vectorspace E⟩ ..
  {
    assume l: ?L
    show ?R
    proof
      fix x assume x: x ∈ H
      have h x ≤ |h x| by arith
      also from l x have ... ≤ p x ..
      finally show h x ≤ p x .
    qed
  }
next
  assume r: ?R
  show ?L

```

```

proof
  fix  $x$  assume  $x: x \in H$ 
  show  $\bigwedge a b :: \text{real}. - a \leq b \implies b \leq a \implies |b| \leq a$ 
    by arith
  from  $\langle \text{linearform } H \ h \rangle$  and  $H \ x$ 
  have  $- h \ x = h \ (- x)$  by  $(\text{rule linearform.neg } [\text{symmetric}])$ 
  also
  from  $H \ x$  have  $- x \in H$  by  $(\text{rule vectorspace.neg-closed})$ 
  with  $r$  have  $h \ (- x) \leq p \ (- x)$  ..
  also have  $\dots = p \ x$ 
    using  $\langle \text{seminorm } E \ p \rangle \langle \text{vectorspace } E \rangle$ 
  proof  $(\text{rule seminorm.minus})$ 
    from  $x$  show  $x \in E$  ..
  qed
  finally have  $- h \ x \leq p \ x$  .
  then show  $- p \ x \leq h \ x$  by simp
  from  $r \ x$  show  $h \ x \leq p \ x$  ..
qed
}
qed

end

```

11 Extending non-maximal functions

```

theory HahnBanachExtLemmas
imports FunctionNorm
begin

```

In this section the following context is presumed. Let E be a real vector space with a seminorm q on E . F is a subspace of E and f a linear function on F . We consider a subspace H of E that is a superspace of F and a linear form h on H . H is not equal to E and x_0 is an element in $E - H$. H is extended to the direct sum $H' = H + \text{lin } x_0$, so for any $x \in H'$ the decomposition of $x = y + a \cdot x_0$ with $y \in H$ is unique. h' is defined on H' by $h' \ x = h \ y + a \cdot \xi$ for a certain ξ .

Subsequently we show some properties of this extension h' of h .

This lemma will be used to show the existence of a linear extension of f (see page 48). It is a consequence of the completeness of \mathbb{R} . To show

$$\exists \xi. \forall y \in F. a \ y \leq \xi \wedge \xi \leq b \ y$$

it suffices to show that

$$\forall u \in F. \forall v \in F. a \ u \leq b \ v$$

```

lemma ex-xi:
  assumes vectorspace  $F$ 
  assumes  $r: \bigwedge u \ v. u \in F \implies v \in F \implies a \ u \leq b \ v$ 
  shows  $\exists xi :: \text{real}. \forall y \in F. a \ y \leq xi \wedge xi \leq b \ y$ 
proof -

```


interpret *vectorspace* F **by** *fact*

From the completeness of the reals follows: The set $S = \{a \mid u. u \in F\}$ has a supremum, if it is non-empty and has an upper bound.

```

let ?S = {a u | u. u ∈ F}
have ∃ xi. lub ?S xi
proof (rule real-complete)
  have a 0 ∈ ?S by blast
  then show ∃ X. X ∈ ?S ..
  have ∀ y ∈ ?S. y ≤ b 0
  proof
    fix y assume y: y ∈ ?S
    then obtain u where u: u ∈ F and y: y = a u by blast
    from u and zero have a u ≤ b 0 by (rule r)
    with y show y ≤ b 0 by (simp only:)
  qed
  then show ∃ u. ∀ y ∈ ?S. y ≤ u ..
qed
then obtain xi where xi: lub ?S xi ..
{
  fix y assume y ∈ F
  then have a y ∈ ?S by blast
  with xi have a y ≤ xi by (rule lub.upper)
} moreover {
  fix y assume y: y ∈ F
  from xi have xi ≤ b y
  proof (rule lub.least)
    fix au assume au ∈ ?S
    then obtain u where u: u ∈ F and au: au = a u by blast
    from u y have a u ≤ b y by (rule r)
    with au show au ≤ b y by (simp only:)
  qed
} ultimately show ∃ xi. ∀ y ∈ F. a y ≤ xi ∧ xi ≤ b y by blast
qed

```

The function h' is defined as a $h' x = h y + a \cdot \xi$ where $x = y + a \cdot \xi$ is a linear extension of h to H' .

lemma h' -lf:

```

assumes h'-def:  $h' \equiv \lambda x. \text{let } (y, a) =$ 
  SOME (y, a).  $x = y + a \cdot x0 \wedge y \in H \text{ in } h y + a * xi$ 
and H'-def:  $H' \equiv H + \text{lin } x0$ 
and HE:  $H \leq E$ 

```

```

assumes linearform H h
assumes x0:  $x0 \notin H \ x0 \in E \ x0 \neq 0$ 
assumes E: vectorspace E
shows linearform H' h'

```

proof –

```

interpret linearform H h by fact
interpret vectorspace E by fact
show ?thesis
proof
  note E = ⟨vectorspace E⟩
  have H': vectorspace H'

```

```

proof (unfold  $H'$ -def)
  from  $\langle x0 \in E \rangle$ 
  have  $\text{lin } x0 \trianglelefteq E$  ..
  with  $HE$  show vectorspace ( $H + \text{lin } x0$ ) using  $E$  ..
qed
{
  fix  $x1\ x2$  assume  $x1: x1 \in H'$  and  $x2: x2 \in H'$ 
  show  $h'(x1 + x2) = h' x1 + h' x2$ 
  proof -
    from  $H'$   $x1\ x2$  have  $x1 + x2 \in H'$ 
    by (rule vectorspace.add-closed)
    with  $x1\ x2$  obtain  $y\ y1\ y2\ a\ a1\ a2$  where
       $x1x2: x1 + x2 = y + a \cdot x0$  and  $y: y \in H$ 
      and  $x1\text{-rep}: x1 = y1 + a1 \cdot x0$  and  $y1: y1 \in H$ 
      and  $x2\text{-rep}: x2 = y2 + a2 \cdot x0$  and  $y2: y2 \in H$ 
      unfolding  $H'$ -def sum-def lin-def by blast

    have  $ya: y1 + y2 = y \wedge a1 + a2 = a$  using  $E\ HE - y\ x0$ 
    proof (rule decomp- $H'$ ) from  $HE\ y1\ y2$  show  $y1 + y2 \in H$ 
      by (rule subspace.add-closed)
    from  $x0$  and  $HE\ y1\ y2$ 
    have  $x0 \in E\ y \in E\ y1 \in E\ y2 \in E$  by auto
    with  $x1\text{-rep}\ x2\text{-rep}$  have  $(y1 + y2) + (a1 + a2) \cdot x0 = x1 + x2$ 
      by (simp add: add-ac add-mult-distrib2)
    also note  $x1x2$ 
    finally show  $(y1 + y2) + (a1 + a2) \cdot x0 = y + a \cdot x0$  .
  qed

  from  $h'\text{-def}\ x1x2\ E\ HE\ y\ x0$ 
  have  $h'(x1 + x2) = h\ y + a * xi$ 
    by (rule h'-definite)
  also have  $\dots = h\ (y1 + y2) + (a1 + a2) * xi$ 
    by (simp only: ya)
  also from  $y1\ y2$  have  $h\ (y1 + y2) = h\ y1 + h\ y2$ 
    by simp
  also have  $\dots + (a1 + a2) * xi = (h\ y1 + a1 * xi) + (h\ y2 + a2 * xi)$ 
    by (simp add: left-distrib)
  also from  $h'\text{-def}\ x1\text{-rep}\ E\ HE\ y1\ x0$ 
  have  $h\ y1 + a1 * xi = h' x1$ 
    by (rule h'-definite [symmetric])
  also from  $h'\text{-def}\ x2\text{-rep}\ E\ HE\ y2\ x0$ 
  have  $h\ y2 + a2 * xi = h' x2$ 
    by (rule h'-definite [symmetric])
  finally show ?thesis .
qed
next
fix  $x1\ c$  assume  $x1: x1 \in H'$ 
show  $h'(c \cdot x1) = c * (h' x1)$ 
proof -
  from  $H'$   $x1$  have  $ax1: c \cdot x1 \in H'$ 
    by (rule vectorspace.mult-closed)
  with  $x1$  obtain  $y\ a\ y1\ a1$  where
     $cx1\text{-rep}: c \cdot x1 = y + a \cdot x0$  and  $y: y \in H$ 
    and  $x1\text{-rep}: x1 = y1 + a1 \cdot x0$  and  $y1: y1 \in H$ 

```

```

    unfolding  $H'$ -def sum-def lin-def by blast

  have ya:  $c \cdot y1 = y \wedge c * a1 = a$  using  $E$  HE -  $y$   $x0$ 
  proof (rule decomp- $H'$ )
    from HE  $y1$  show  $c \cdot y1 \in H$ 
      by (rule subspace.mult-closed)
    from  $x0$  and HE  $y$   $y1$ 
    have  $x0 \in E$   $y \in E$   $y1 \in E$  by auto
    with  $x1$ -rep have  $c \cdot y1 + (c * a1) \cdot x0 = c \cdot x1$ 
      by (simp add: mult-assoc add-mult-distrib1)
    also note  $cx1$ -rep
    finally show  $c \cdot y1 + (c * a1) \cdot x0 = y + a \cdot x0$  .
  qed

  from  $h'$ -def  $cx1$ -rep  $E$  HE  $y$   $x0$  have  $h' (c \cdot x1) = h y + a * xi$ 
    by (rule  $h'$ -definite)
  also have  $\dots = h (c \cdot y1) + (c * a1) * xi$ 
    by (simp only: ya)
  also from  $y1$  have  $h (c \cdot y1) = c * h y1$ 
    by simp
  also have  $\dots + (c * a1) * xi = c * (h y1 + a1 * xi)$ 
    by (simp only: right-distrib)
  also from  $h'$ -def  $x1$ -rep  $E$  HE  $y1$   $x0$  have  $h y1 + a1 * xi = h' x1$ 
    by (rule  $h'$ -definite [symmetric])
  finally show ?thesis .
}
qed
qed

```

The linear extension h' of h is bounded by the seminorm p .

lemma h' -norm-pres:

```

  assumes  $h'$ -def:  $h' \equiv \lambda x. \text{let } (y, a) =$ 
    SOME  $(y, a). x = y + a \cdot x0 \wedge y \in H \text{ in } h y + a * xi$ 
    and  $H'$ -def:  $H' \equiv H + \text{lin } x0$ 
    and  $x0: x0 \notin H$   $x0 \in E$   $x0 \neq 0$ 
  assumes  $E$ : vectorspace  $E$  and HE: subspace  $H$   $E$ 
    and seminorm  $E$   $p$  and linearform  $H$   $h$ 
  assumes  $a$ :  $\forall y \in H. h y \leq p y$ 
    and  $a'$ :  $\forall y \in H. -p (y + x0) - h y \leq xi \wedge xi \leq p (y + x0) - h y$ 
  shows  $\forall x \in H'. h' x \leq p x$ 

```

proof –

```

  interpret vectorspace  $E$  by fact
  interpret subspace  $H$   $E$  by fact
  interpret seminorm  $E$   $p$  by fact
  interpret linearform  $H$   $h$  by fact
  show ?thesis

```

proof

```

  fix  $x$  assume  $x'$ :  $x \in H'$ 

```

```

  show  $h' x \leq p x$ 

```

proof –

```

  from  $a'$  have  $a1: \forall ya \in H. -p (ya + x0) - h ya \leq xi$ 
    and  $a2: \forall ya \in H. xi \leq p (ya + x0) - h ya$  by auto
  from  $x'$  obtain  $y$   $a$  where

```

```

    x-rep:  $x = y + a \cdot x0$  and  $y: y \in H$ 
    unfolding  $H'$ -def sum-def lin-def by blast
    from  $y$  have  $y': y \in E$  ..
    from  $y$  have  $ay: \text{inverse } a \cdot y \in H$  by simp

    from  $h'$ -def x-rep  $E HE y x0$  have  $h' x = h y + a * xi$ 
    by (rule  $h'$ -definite)
    also have  $\dots \leq p (y + a \cdot x0)$ 
    proof (rule linorder-cases)
      assume  $z: a = 0$ 
      then have  $h y + a * xi = h y$  by simp
      also from  $a y$  have  $\dots \leq p y$  ..
      also from  $x0 y' z$  have  $p y = p (y + a \cdot x0)$  by simp
      finally show ?thesis .
    next

```

In the case $a < 0$, we use a_1 with ya taken as y / a :

```

    assume  $lz: a < 0$  then have  $nz: a \neq 0$  by simp
    from  $a1 ay$ 
    have  $- p (\text{inverse } a \cdot y + x0) - h (\text{inverse } a \cdot y) \leq xi$  ..
    with  $lz$  have  $a * xi \leq$ 
       $a * (- p (\text{inverse } a \cdot y + x0) - h (\text{inverse } a \cdot y))$ 
    by (simp add: mult-left-mono-neg order-less-imp-le)

    also have  $\dots =$ 
       $- a * (p (\text{inverse } a \cdot y + x0)) - a * (h (\text{inverse } a \cdot y))$ 
    by (simp add: right-diff-distrib)
    also from  $lz x0 y'$  have  $- a * (p (\text{inverse } a \cdot y + x0)) =$ 
       $p (a \cdot (\text{inverse } a \cdot y + x0))$ 
    by (simp add: abs-homogenous)
    also from  $nz x0 y'$  have  $\dots = p (y + a \cdot x0)$ 
    by (simp add: add-mult-distrib1 mult-assoc [symmetric])
    also from  $nz y$  have  $a * (h (\text{inverse } a \cdot y)) = h y$ 
    by simp
    finally have  $a * xi \leq p (y + a \cdot x0) - h y$  .
    then show ?thesis by simp
  next

```

In the case $a > 0$, we use a_2 with ya taken as y / a :

```

    assume  $gz: 0 < a$  then have  $nz: a \neq 0$  by simp
    from  $a2 ay$ 
    have  $xi \leq p (\text{inverse } a \cdot y + x0) - h (\text{inverse } a \cdot y)$  ..
    with  $gz$  have  $a * xi \leq$ 
       $a * (p (\text{inverse } a \cdot y + x0) - h (\text{inverse } a \cdot y))$ 
    by simp
    also have  $\dots = a * p (\text{inverse } a \cdot y + x0) - a * h (\text{inverse } a \cdot y)$ 
    by (simp add: right-diff-distrib)
    also from  $gz x0 y'$ 
    have  $a * p (\text{inverse } a \cdot y + x0) = p (a \cdot (\text{inverse } a \cdot y + x0))$ 
    by (simp add: abs-homogenous)
    also from  $nz x0 y'$  have  $\dots = p (y + a \cdot x0)$ 
    by (simp add: add-mult-distrib1 mult-assoc [symmetric])
    also from  $nz y$  have  $a * h (\text{inverse } a \cdot y) = h y$ 
    by simp

```

```

    finally have  $a * xi \leq p (y + a \cdot x0) - h y$  .
    then show ?thesis by simp
  qed
  also from x-rep have  $\dots = p x$  by (simp only :)
  finally show ?thesis .
qed
qed
qed
end

```

Part III

The Main Proof

12 The Hahn-Banach Theorem

theory *HahnBanach*
imports *HahnBanachLemmas*
begin

We present the proof of two different versions of the Hahn-Banach Theorem, closely following [1, §36].

12.1 The Hahn-Banach Theorem for vector spaces

Hahn-Banach Theorem. Let F be a subspace of a real vector space E , let p be a semi-norm on E , and f be a linear form defined on F such that f is bounded by p , i.e. $\forall x \in F. f x \leq p x$. Then f can be extended to a linear form h on E such that h is norm-preserving, i.e. h is also bounded by p .

Proof Sketch.

1. Define M as the set of norm-preserving extensions of f to subspaces of E . The linear forms in M are ordered by domain extension.
2. We show that every non-empty chain in M has an upper bound in M .
3. With Zorn's Lemma we conclude that there is a maximal function g in M .
4. The domain H of g is the whole space E , as shown by classical contradiction:
 - Assuming g is not defined on whole E , it can still be extended in a norm-preserving way to a super-space H' of H .
 - Thus g can not be maximal. Contradiction!

theorem *HahnBanach*:

assumes E : *vectorspace* E **and** *subspace* $F E$

and *seminorm* $E p$ **and** *linearform* $F f$

assumes fp : $\forall x \in F. f x \leq p x$

shows $\exists h. \text{linearform } E h \wedge (\forall x \in F. h x = f x) \wedge (\forall x \in E. h x \leq p x)$

— Let E be a vector space, F a subspace of E , p a seminorm on E ,

— and f a linear form on F such that f is bounded by p ,

— then f can be extended to a linear form h on E in a norm-preserving way.

proof —

interpret *vectorspace* E **by** *fact*

interpret *subspace* $F E$ **by** *fact*

interpret *seminorm* $E p$ **by** *fact*

interpret *linearform* $F f$ **by** *fact*

def $M \equiv \text{norm-pres-extensions } E p F f$

then have M : $M = \dots$ **by** (*simp only*:)

```

from  $E$  have  $F$ : vectorspace  $F$  ..
note  $FE = \langle F \trianglelefteq E \rangle$ 
{
  fix  $c$  assume  $cM$ :  $c \in \text{chain } M$  and  $ex$ :  $\exists x. x \in c$ 
  have  $\bigcup c \in M$ 
    — Show that every non-empty chain  $c$  of  $M$  has an upper bound in  $M$ :
    —  $\bigcup c$  is greater than any element of the chain  $c$ , so it suffices to show  $\bigcup c \in M$ .
    unfolding  $M\text{-def}$ 
  proof (rule norm-pres-extensionI)
    let  $?H = \text{domain } (\bigcup c)$ 
    let  $?h = \text{funct } (\bigcup c)$ 

    have  $a$ : graph  $?H$   $?h = \bigcup c$ 
    proof (rule graph-domain-funct)
      fix  $x y z$  assume  $(x, y) \in \bigcup c$  and  $(x, z) \in \bigcup c$ 
      with  $M\text{-def } cM$  show  $z = y$  by (rule sup-definite)
    qed
    moreover from  $M$   $cM$   $a$  have linearform  $?H$   $?h$ 
      by (rule sup-lf)
    moreover from  $a$   $M$   $cM$   $ex$   $FE$   $E$  have  $?H \trianglelefteq E$ 
      by (rule sup-subE)
    moreover from  $a$   $M$   $cM$   $ex$   $FE$  have  $F \trianglelefteq ?H$ 
      by (rule sup-supF)
    moreover from  $a$   $M$   $cM$   $ex$  have graph  $F$   $f \subseteq \text{graph } ?H$   $?h$ 
      by (rule sup-ext)
    moreover from  $a$   $M$   $cM$  have  $\forall x \in ?H. ?h x \leq p x$ 
      by (rule sup-norm-pres)
    ultimately show  $\exists H h. \bigcup c = \text{graph } H$   $h$ 
       $\wedge$  linearform  $H$   $h$ 
       $\wedge$   $H \trianglelefteq E$ 
       $\wedge$   $F \trianglelefteq H$ 
       $\wedge$  graph  $F$   $f \subseteq \text{graph } H$   $h$ 
       $\wedge$  ( $\forall x \in H. h x \leq p x$ ) by blast
    qed
  }
then have  $\exists g \in M. \forall x \in M. g \subseteq x \longrightarrow g = x$ 
  — With Zorn's Lemma we can conclude that there is a maximal element in  $M$ .

  proof (rule Zorn's-Lemma)
    — We show that  $M$  is non-empty:
    show graph  $F$   $f \in M$ 
      unfolding  $M\text{-def}$ 
    proof (rule norm-pres-extensionI2)
      show linearform  $F$   $f$  by fact
      show  $F \trianglelefteq E$  by fact
      from  $F$  show  $F \trianglelefteq F$  by (rule vectorspace.subspace-refl)
      show graph  $F$   $f \subseteq \text{graph } F$   $f$  ..
      show  $\forall x \in F. f x \leq p x$  by fact
    qed
  qed
  then obtain  $g$  where  $gM$ :  $g \in M$  and  $gx$ :  $\forall x \in M. g \subseteq x \longrightarrow g = x$ 
    by blast
  from  $gM$  obtain  $H$   $h$  where
     $g\text{-rep}$ :  $g = \text{graph } H$   $h$ 
    and linearform: linearform  $H$   $h$ 

```

and $HE: H \trianglelefteq E$ **and** $FH: F \trianglelefteq H$
and $graphs: graph\ F\ f \subseteq graph\ H\ h$
and $hp: \forall x \in H. h\ x \leq p\ x$ **unfolding** $M-def$..
 — g is a norm-preserving extension of f , in other words:
 — g is the graph of some linear form h defined on a subspace H of E ,
 — and h is an extension of f that is again bounded by p .
from $HE\ E$ **have** $H: vectorspace\ H$
by (*rule subspace.vectorspace*)

have $HE-eq: H = E$
 — We show that h is defined on whole E by classical contradiction.
proof (*rule classical*)
assume $neg: H \neq E$
 — Assume h is not defined on whole E . Then show that h can be extended
 — in a norm-preserving way to a function h' with the graph g' .
have $\exists g' \in M. g \subseteq g' \wedge g \neq g'$
proof —
from HE **have** $H \subseteq E$..
with neg **obtain** x' **where** $x'E: x' \in E$ **and** $x' \notin H$ **by** *blast*
obtain $x': x' \neq 0$
proof
show $x' \neq 0$
proof
assume $x' = 0$
with H **have** $x' \in H$ **by** (*simp only: vectorspace.zero*)
with $\langle x' \notin H \rangle$ **show** *False* **by** *contradiction*
qed
qed

def $H' \equiv H + \text{lin } x'$
 — Define H' as the direct sum of H and the linear closure of x' .
have $HH': H \trianglelefteq H'$
proof (*unfold H'-def*)
from $x'E$ **have** $vectorspace\ (\text{lin } x')$..
with H **show** $H \trianglelefteq H + \text{lin } x'$..
qed

obtain xi **where**
 $xi: \forall y \in H. -p\ (y + x') - h\ y \leq xi$
 $\wedge xi \leq p\ (y + x') - h\ y$
 — Pick a real number ξ that fulfills certain inequations; this will
 — be used to establish that h' is a norm-preserving extension of h .
proof —
from H **have** $\exists xi. \forall y \in H. -p\ (y + x') - h\ y \leq xi$
 $\wedge xi \leq p\ (y + x') - h\ y$
proof (*rule ex-xi*)
fix $u\ v$ **assume** $u: u \in H$ **and** $v: v \in H$
with HE **have** $uE: u \in E$ **and** $vE: v \in E$ **by** *auto*
from $H\ u\ v$ **linearform** **have** $h\ v - h\ u = h\ (v - u)$
by (*simp add: linearform.diff*)
also from hp **and** $H\ u\ v$ **have** $\dots \leq p\ (v - u)$
by (*simp only: vectorspace.diff-closed*)
also from $x'E\ uE\ vE$ **have** $v - u = x' + -\ x' + v + -\ u$


```

    by (simp add: diff-eq1)
  also from  $x'E uE vE$  have  $\dots = v + x' + -(u + x')$ 
    by (simp add: add-ac)
  also from  $x'E uE vE$  have  $\dots = (v + x') - (u + x')$ 
    by (simp add: diff-eq1)
  also from  $x'E uE vE E$  have  $p \dots \leq p (v + x') + p (u + x')$ 
    by (simp add: diff-subadditive)
  finally have  $h v - h u \leq p (v + x') + p (u + x')$  .
  then show  $-p (u + x') - h u \leq p (v + x') - h v$  by simp
qed
then show thesis by (blast intro: that)
qed

def h'  $\equiv \lambda x. \text{let } (y, a) =$ 
  SOME  $(y, a). x = y + a \cdot x' \wedge y \in H \text{ in } h y + a \cdot x'$ 
  — Define the extension  $h'$  of  $h$  to  $H'$  using  $\xi$ .

have  $g \subseteq \text{graph } H' h' \wedge g \neq \text{graph } H' h'$ 
  —  $h'$  is an extension of  $h \dots$ 
proof
  show  $g \subseteq \text{graph } H' h'$ 
  proof —
    have  $\text{graph } H h \subseteq \text{graph } H' h'$ 
    proof (rule graph-extI)
      fix  $t$  assume  $t: t \in H$ 
      from  $E HE t$  have  $(\text{SOME } (y, a). t = y + a \cdot x' \wedge y \in H) = (t, 0)$ 
        using  $\langle x' \notin H \rangle \langle x' \in E \rangle \langle x' \neq 0 \rangle$  by (rule decomp- $H'-H$ )
      with  $h'$ -def show  $h t = h' t$  by (simp add: Let-def)
    next
      from  $HH'$  show  $H \subseteq H' ..$ 
    qed
    with  $g$ -rep show ?thesis by (simp only:)
  qed

show  $g \neq \text{graph } H' h'$ 
proof —
  have  $\text{graph } H h \neq \text{graph } H' h'$ 
  proof
    assume eq:  $\text{graph } H h = \text{graph } H' h'$ 
    have  $x' \in H'$ 
      unfolding  $H'$ -def
    proof
      from  $H$  show  $0 \in H$  by (rule vectorspace.zero)
      from  $x'E$  show  $x' \in \text{lin } x'$  by (rule x-lin-x)
      from  $x'E$  show  $x' = 0 + x'$  by simp
    qed
    then have  $(x', h' x') \in \text{graph } H' h' ..$ 
    with eq have  $(x', h' x') \in \text{graph } H h$  by (simp only:)
    then have  $x' \in H ..$ 
    with  $\langle x' \notin H \rangle$  show False by contradiction
  qed
  with  $g$ -rep show ?thesis by simp
qed
qed

```

moreover have $\text{graph } H' h' \in M$
 — and h' is norm-preserving.
proof (*unfold M-def*)
 show $\text{graph } H' h' \in \text{norm-pres-extensions } E p F f$
proof (*rule norm-pres-extensionI2*)
 show *linearform* $H' h'$
 using $h'\text{-def } H'\text{-def } HE$ *linearform* $\langle x' \notin H \rangle \langle x' \in E \rangle \langle x' \neq 0 \rangle E$
 by (*rule h'-lf*)
 show $H' \trianglelefteq E$
 unfolding $H'\text{-def}$
proof
 show $H \trianglelefteq E$ **by fact**
 show *vectorspace* E **by fact**
 from $x'E$ show $\text{lin } x' \trianglelefteq E$..
qed
 from $H \langle F \trianglelefteq H \rangle HH'$ show $FH': F \trianglelefteq H'$
 by (*rule vectorspace.subspace-trans*)
 show $\text{graph } F f \subseteq \text{graph } H' h'$
proof (*rule graph-extI*)
 fix x **assume** $x: x \in F$
 with *graphs* **have** $f x = h x$..
 also **have** $\dots = h x + 0 * xi$ **by** *simp*
 also **have** $\dots = (\text{let } (y, a) = (x, 0) \text{ in } h y + a * xi)$
 by (*simp add: Let-def*)
 also **have** $(x, 0) =$
 $(\text{SOME } (y, a). x = y + a \cdot x' \wedge y \in H)$
 using $E HE$
proof (*rule decomp-H'-H [symmetric]*)
 from $FH x$ **show** $x \in H$..
 from x' **show** $x' \neq 0$.
 show $x' \notin H$ **by fact**
 show $x' \in E$ **by fact**
qed
 also **have**
 $(\text{let } (y, a) = (\text{SOME } (y, a). x = y + a \cdot x' \wedge y \in H)$
 $\text{in } h y + a * xi) = h' x$ **by** (*simp only: h'-def*)
 finally **show** $f x = h' x$.
next
 from FH' **show** $F \subseteq H'$..
qed
 show $\forall x \in H'. h' x \leq p x$
 using $h'\text{-def } H'\text{-def } \langle x' \notin H \rangle \langle x' \in E \rangle \langle x' \neq 0 \rangle E HE$
 $\langle \text{seminorm } E p \rangle$ *linearform* **and** $hp xi$
 by (*rule h'-norm-pres*)
qed
qed
 ultimately **show** *?thesis* ..
qed
 then **have** $\neg (\forall x \in M. g \subseteq x \longrightarrow g = x)$ **by** *simp*
 — So the graph g of h cannot be maximal. Contradiction!
 with gx **show** $H = E$ **by** *contradiction*
qed

from $HE\text{-eq}$ and *linearform* **have** *linearform* $E h$

```

  by (simp only:)
  moreover have  $\forall x \in F. h\ x = f\ x$ 
  proof
    fix x assume  $x \in F$ 
    with graphs have  $f\ x = h\ x$  ..
    then show  $h\ x = f\ x$  ..
  qed
  moreover from HE-eq and hp have  $\forall x \in E. h\ x \leq p\ x$ 
  by (simp only:)
  ultimately show ?thesis by blast
qed

```

12.2 Alternative formulation

The following alternative formulation of the Hahn-Banach Theorem uses the fact that for a real linear form f and a seminorm p the following inequations are equivalent:¹

$$\forall x \in H. |h\ x| \leq p\ x \quad \text{and} \quad \forall x \in H. h\ x \leq p\ x$$

```

theorem abs-HahnBanach:
  assumes E: vectorspace E and FE: subspace F E
    and lf: linearform F f and sn: seminorm E p
  assumes fp:  $\forall x \in F. |f\ x| \leq p\ x$ 
  shows  $\exists g. \text{linearform } E\ g$ 
     $\wedge (\forall x \in F. g\ x = f\ x)$ 
     $\wedge (\forall x \in E. |g\ x| \leq p\ x)$ 
  proof -
    interpret vectorspace E by fact
    interpret subspace F E by fact
    interpret linearform F f by fact
    interpret seminorm E p by fact
    have  $\exists g. \text{linearform } E\ g \wedge (\forall x \in F. g\ x = f\ x) \wedge (\forall x \in E. g\ x \leq p\ x)$ 
      using E FE sn lf
    proof (rule HahnBanach)
      show  $\forall x \in F. f\ x \leq p\ x$ 
        using FE E sn lf and fp by (rule abs-ineq-iff [THEN iffD1])
    qed
    then obtain g where lg: linearform E g and *:  $\forall x \in F. g\ x = f\ x$ 
      and **:  $\forall x \in E. g\ x \leq p\ x$  by blast
    have  $\forall x \in E. |g\ x| \leq p\ x$ 
      using - E sn lg **
    proof (rule abs-ineq-iff [THEN iffD2])
      show  $E \sqsubseteq E$  ..
    qed
    with lg * show ?thesis by blast
  qed

```

12.3 The Hahn-Banach Theorem for normed spaces

Every continuous linear form f on a subspace F of a norm space E , can be extended to a continuous linear form g on E such that $\|f\| = \|g\|$.

¹This was shown in lemma *abs-ineq-iff* (see page 39).

theorem *norm-HahnBanach*:
fixes V **and** *norm* ($\|\cdot\|$)
fixes B **defines** $\bigwedge V f. B V f \equiv \{0\} \cup \{\|f x\| / \|x\| \mid x. x \neq 0 \wedge x \in V\}$
fixes *fn-norm* ($\|\cdot\|$ -- $[0, 1000]$ 999)
defines $\bigwedge V f. \|f\|$ - $V \equiv \bigsqcup (B V f)$
assumes *E-norm*: *normed-vectorspace* E *norm* **and** *FE*: *subspace* $F E$
and *linearform*: *linearform* $F f$ **and** *continuous F norm f*
shows $\exists g. \text{linearform } E g$
 $\wedge \text{continuous } E \text{ norm } g$
 $\wedge (\forall x \in F. g x = f x)$
 $\wedge \|g\|$ - $E = \|f\|$ - F
proof –
interpret *normed-vectorspace* E *norm* **by** *fact*
interpret *normed-vectorspace-with-fn-norm* E *norm* B *fn-norm*
by (*auto simp*: B -*def fn-norm-def*) *intro-locales*
interpret *subspace* $F E$ **by** *fact*
interpret *linearform* $F f$ **by** *fact*
interpret *continuous F norm f* **by** *fact*
have E : *vectorspace* E **by** *intro-locales*
have F : *vectorspace* F **by** *rule intro-locales*
have F -*norm*: *normed-vectorspace* F *norm*
using FE E -*norm* **by** (*rule subspace-normed-vs*)
have *ge-zero*: $0 \leq \|f\|$ - F
by (*rule normed-vectorspace-with-fn-norm.fn-norm-ge-zero*
 $[OF \text{ normed-vectorspace-with-fn-norm.intro,}$
 $OF F\text{-norm (continuous } F \text{ norm } f), \text{ folded } B\text{-def fn-norm-def}]$)

We define a function p on E as follows: $p x = \|f\| \cdot \|x\|$

def $p \equiv \lambda x. \|f\|$ - $F * \|x\|$

p is a seminorm on E :

have q : *seminorm* $E p$

proof

fix $x y a$ **assume** $x: x \in E$ **and** $y: y \in E$

p is positive definite:

have $0 \leq \|f\|$ - F **by** (*rule ge-zero*)
moreover from x **have** $0 \leq \|x\|$..
ultimately show $0 \leq p x$
by (*simp add*: p -*def zero-le-mult-iff*)

p is absolutely homogenous:

show $p (a \cdot x) = |a| * p x$
proof –
have $p (a \cdot x) = \|f\|$ - $F * \|a \cdot x\|$ **by** (*simp only*: p -*def*)
also from x **have** $\|a \cdot x\| = |a| * \|x\|$ **by** (*rule abs-homogenous*)
also have $\|f\|$ - $F * (|a| * \|x\|) = |a| * (\|f\|$ - $F * \|x\|)$ **by** *simp*
also have $\dots = |a| * p x$ **by** (*simp only*: p -*def*)
finally show *?thesis* .
qed

Furthermore, p is subadditive:

show $p (x + y) \leq p x + p y$

```

proof –
  have  $p(x + y) = \|f\|{-}F * \|x + y\|$  by (simp only: p-def)
  also have  $a: 0 \leq \|f\|{-}F$  by (rule ge-zero)
  from  $x\ y$  have  $\|x + y\| \leq \|x\| + \|y\|$  ..
  with  $a$  have  $\|f\|{-}F * \|x + y\| \leq \|f\|{-}F * (\|x\| + \|y\|)$ 
    by (simp add: mult-left-mono)
  also have  $\dots = \|f\|{-}F * \|x\| + \|f\|{-}F * \|y\|$  by (simp only: right-distrib)
  also have  $\dots = p\ x + p\ y$  by (simp only: p-def)
  finally show ?thesis .
qed
qed

```

f is bounded by p .

```

have  $\forall x \in F. |f\ x| \leq p\ x$ 
proof
  fix  $x$  assume  $x \in F$ 
  with  $\langle \text{continuous } F \text{ norm } f \rangle$  and linearform
  show  $|f\ x| \leq p\ x$ 
    unfolding p-def by (rule normed-vectorspace-with-fn-norm.fn-norm-le-cong
      [OF normed-vectorspace-with-fn-norm.intro,
       OF F-norm, folded B-def fn-norm-def])
qed

```

Using the fact that p is a seminorm and f is bounded by p we can apply the Hahn-Banach Theorem for real vector spaces. So f can be extended in a norm-preserving way to some function g on the whole vector space E .

```

with  $E\ FE$  linearform  $q$  obtain  $g$  where
  linearformE: linearform E g
  and  $a: \forall x \in F. g\ x = f\ x$ 
  and  $b: \forall x \in E. |g\ x| \leq p\ x$ 
  by (rule abs-HahnBanach [elim-format]) iprover

```

We furthermore have to show that g is also continuous:

```

have g-cont: continuous E norm g using linearformE
proof
  fix  $x$  assume  $x \in E$ 
  with  $b$  show  $|g\ x| \leq \|f\|{-}F * \|x\|$ 
    by (simp only: p-def)
qed

```

To complete the proof, we show that $\|g\| = \|f\|$.

```

have  $\|g\|{-}E = \|f\|{-}F$ 
proof (rule order-antisym)

```

First we show $\|g\| \leq \|f\|$. The function norm $\|g\|$ is defined as the smallest $c \in \mathbb{R}$ such that

$$\forall x \in E. |g\ x| \leq c \cdot \|x\|$$

Furthermore holds

$$\forall x \in E. |g\ x| \leq \|f\| \cdot \|x\|$$

```

have  $\forall x \in E. |g\ x| \leq \|f\|{-}F * \|x\|$ 

```

```

proof
  fix  $x$  assume  $x \in E$ 
  with  $b$  show  $|g\ x| \leq \|f\| \cdot F * \|x\|$ 
    by (simp only: p-def)
qed
from  $g\text{-cont}$  this  $g\text{-zero}$ 
show  $\|g\| \cdot E \leq \|f\| \cdot F$ 
  by (rule fn-norm-least [of g, folded B-def fn-norm-def])

```

The other direction is achieved by a similar argument.

```

show  $\|f\| \cdot F \leq \|g\| \cdot E$ 
proof (rule normed-vectorspace-with-fn-norm.fn-norm-least
  [OF normed-vectorspace-with-fn-norm.intro,
  OF F-norm, folded B-def fn-norm-def])
show  $\forall x \in F. |f\ x| \leq \|g\| \cdot E * \|x\|$ 
proof
  fix  $x$  assume  $x: x \in F$ 
  from  $a\ x$  have  $g\ x = f\ x$  ..
  then have  $|f\ x| = |g\ x|$  by (simp only:)
  also from  $g\text{-cont}$ 
  have  $\dots \leq \|g\| \cdot E * \|x\|$ 
  proof (rule fn-norm-le-cong [OF - linearformE, folded B-def fn-norm-def])
    from  $FE\ x$  show  $x \in E$  ..
  qed
  finally show  $|f\ x| \leq \|g\| \cdot E * \|x\|$  .
qed
show  $0 \leq \|g\| \cdot E$ 
  using  $g\text{-cont}$ 
  by (rule fn-norm-ge-zero [of g, folded B-def fn-norm-def])
show continuous F norm f by fact
qed
qed
with linearformE a g-cont show ?thesis by blast
qed

end

```

References

- [1] H. Heuser. *Funktionalanalysis: Theorie und Anwendung*. Teubner, 1986.
- [2] L. Narici and E. Beckenstein. The Hahn-Banach Theorem: The life and times. In *Topology Atlas*. York University, Toronto, Ontario, Canada, 1996. <http://at.yorku.ca/topology/preprint.htm> and <http://at.yorku.ca/p/a/a/a/16.htm>.
- [3] B. Nowak and A. Trybulec. Hahn-Banach theorem. *Journal of Formalized Mathematics*, 5, 1993. <http://mizar.uwb.edu.pl/JFM/Vol5/hahnban.html>.