The Lévy-Prokhorov Metric

Michikazu Hirata

June 11, 2024

Abstract

We formalize the Lévy-Prokhorov metric, a metric on finite measures, mainly following the lecture notes by Gaans [4]. This entry includes the following formalization.

- Characterizations of closed sets, open sets, and topology by limit.
- A special case of Alaoglu's theorem.
- Weak convergence and the Portmanteau theorem.
- The Lévy-Prokhorov metric and its completeness and separability.
- The equivalence of the topology of weak convergence and the topology generated by the Lévy-Prokhorov metric.
- Prokhorov's theorem.
- Equality of two σ-algebras on the space of finite measures. One is the Borel algebra of the Lévy-Prokhorov metric and the other is the least σ-algebra that makes $(\lambda\mu.\ \mu(A))$ measurable for all measurable sets A.
- The space of finite measures on a standard Borel space is also a standard Borel space.

Contents

1 Preliminaries

theory *Lemmas-Levy-Prokhorov*
  imports *Standard-Borel-Spaces.StandardBorel*
begin

lemma(**in** *Metric-space*) [*measurable*]:
  shows *mball-sets*: *mball x e* $\in$ *sets* (*borel-of mtopology*)
  and *mcball-sets*: *mcball x e* $\in$ *sets* (*borel-of mtopology*)
  by(*auto simp*: *borel-of-open borel-of-closed*)

lemma *Metric-space-eq-MCauchy*:
  assumes *Metric-space M d* $\bigwedge x\ y.\ x \in M \Longrightarrow y \in M \Longrightarrow d\ x\ y = d'\ x\ y$
    and $\bigwedge x\ y.\ d'\ x\ y = d'\ y\ x$ $\bigwedge x\ y.\ d'\ x\ y \geq 0$
  shows *Metric-space.MCauchy M d xn* $\longleftrightarrow$ *Metric-space.MCauchy M d' xn*
proof −
  interpret *d*: *Metric-space M d* **by** *fact*

interpret *d'*: *Metric-space M d'*
using *Metric-space-eq assms d.Metric-space-axioms* **by** *blast*
show *?thesis*
using *assms*(*2*) **by**(*auto simp*: *d.MCauchy-def d'.MCauchy-def subsetD*)
qed

lemma *borel-of-compact*: *Hausdorff-space X* $\Longrightarrow$ *compactin X K* $\Longrightarrow$ *K* $\in$ *sets* (*borel-of X*)
by(*auto intro*!: *borel-of-closed compactin-imp-closedin*)

lemma *prob-algebra-cong*: *sets M* = *sets N* $\Longrightarrow$ *prob-algebra M* = *prob-algebra N*
by(*simp add*: *prob-algebra-def cong*: *subprob-algebra-cong*)

lemma *topology-eq-closedin*: *X* = *Y* $\longleftrightarrow$ ($\forall$ *C*. *closedin X C* $\longleftrightarrow$ *closedin Y C*)
unfolding *topology-eq*
by (*metis closedin-def closedin-topspace openin-closedin-eq openin-topspace subset-antisym*)

Another version of *finite-measure ?M* $\Longrightarrow$ *countable* {*x*. *Sigma-Algebra.measure ?M* {*x*} $\neq$ *0*}

lemma(**in** *finite-measure*) *countable-support-sets*:
assumes *disjoint-family-on Ai D*
shows *countable* {*i*∈*D*. *measure M* (*Ai i*) $\neq$ *0*}
proof *cases*
assume *measure M* (*space M*) = *0*
with *bounded-measure measure-le-0-iff* **have** [*simp*]:{*i*∈*D*. *measure M* (*Ai i*) $\neq$ *0*} = {}
by *auto*
show *?thesis*
by *simp*
next
let *?M* = *measure M* (*space M*) **and** *?m* = λ*i*. *measure M* (*Ai i*)
assume *?M* $\neq$ *0*
then have ∗: {*i*∈*D*. *?m i* $\neq$ *0*} = ($\bigcup$*n*. {*i*∈*D*. *?M* / *Suc n* < *?m i*})
using *reals-Archimedean*[*of ?m x* / *?M* **for** *x*]
by (*auto simp*: *field-simps not-le*[*symmetric*] *divide-le-0-iff measure-le-0-iff*)
have ∗∗: $\bigwedge$*n*. *finite* {*i*∈*D*. *?M* / *Suc n* < *?m i*}
proof (*rule ccontr*)
fix *n* **assume** *infinite* {*i*∈*D*. *?M* / *Suc n* < *?m i*} (**is** *infinite ?X*)
then obtain *X* **where** *finite X card X* = *Suc* (*Suc n*) *X* $\subseteq$ *?X*
by (*meson infinite-arbitrarily-large*)
from *this*(*3*) **have** ∗: $\bigwedge$*x*. *x* $\in$ *X* $\Longrightarrow$ *?M* / *Suc n* $\leq$ *?m x*
by *auto*
{ fix *i* **assume** *i* $\in$ *X*
from ‹*?M* $\neq$ *0*› ∗[*OF this*] **have** *?m i* $\neq$ *0* **by** (*auto simp*: *field-simps measure-le-0-iff*)
then have *Ai i* $\in$ *sets M* **by** (*auto dest*: *measure-notin-sets*) **}**

note *sets-Ai* = *this*
have *disj*: *disjoint-family-on Ai X*
using ‹*X* ⊆ *?X*› *assms* **by**(*auto simp*: *disjoint-family-on-def*)
have *?M* < (∑*x*∈*X*. *?M* / *Suc n*)
using ‹*?M* ≠ *0*›
by (*simp add*: ‹*card X* = *Suc* (*Suc n*)› *field-simps less-le*)
also have ... ≤ (∑*x*∈*X*. *?m x*)
by (*rule sum-mono*) *fact*
also have ... = *measure M* (⋃*i*∈*X*. *Ai i*)
using *sets-Ai* ‹*finite X*› **by** (*intro finite-measure-finite-Union*[*symmetric*,*OF* - - *disj*])
(*auto simp*: *disjoint-family-on-def*)
finally have *?M* < *measure M* (⋃*i*∈*X*. *Ai i*) **.**
moreover have *measure M* (⋃*i*∈*X*. *Ai i*) ≤ *?M*
using *sets-Ai*[*THEN sets.sets-into-space*] **by** (*intro finite-measure-mono*) *auto*
ultimately show *False* **by** *simp*
qed
show *?thesis*
unfolding ∗ **by** (*intro countable-UN countableI-type countable-finite*[*OF* ∗∗])
qed

1.1 Finite Sum of Measures

definition *sum-measure* :: *'b measure* ⇒ *'a set* ⇒ (*'a* ⇒ *'b measure*) ⇒ *'b measure*
where
sum-measure M I Mi ≡ *measure-of* (*space M*) (*sets M*) (λ*A*. ∑ *i*∈*I*. *emeasure* (*Mi i*) *A*)

lemma *sum-measure-cong*:
assumes *sets M* = *sets M'* ⋀*i*. *i* ∈ *I* ⟹ *N i* = *N' i*
shows *sum-measure M I N* = *sum-measure M' I N'*
by(*simp add*: *sum-measure-def assms cong*: *sets-eq-imp-space-eq*)

lemma [*simp*]:
shows *space-sum-measure*: *space* (*sum-measure M I Mi*) = *space M*
and *sets-sum-measure*[*measurable-cong*]: *sets* (*sum-measure M I Mi*) = *sets M*
by(*auto simp*: *sum-measure-def*)

lemma *emeasure-sum-measure*:
assumes [*measurable*]:*A* ∈ *sets M* **and** ⋀*i*. *i* ∈ *I* ⟹ *sets* (*Mi i*) = *sets M*
shows *emeasure* (*sum-measure M I Mi*) *A* = (∑ *i*∈*I*. *Mi i A*)

proof(*rule emeasure-measure-of*[*of - space M sets M*])
show *countably-additive* (*sets* (*sum-measure M I Mi*)) (λ*A*. ∑ *i*∈*I*. *emeasure* (*Mi i*) *A*)
unfolding *sum-measure-def sets.sets-measure-of-eq countably-additive-def*
proof *safe*
fix *Ai* :: *nat* ⇒ -
assume *h*:*range Ai* ⊆ *sets M disjoint-family Ai*

then have [*measurable*]: $\bigwedge i\ j.\ j \in I \Longrightarrow Ai\ i \in sets\ (Mi\ j)$
by(*auto simp*: *assms*)
show $(\sum i.\ \sum j{\in}I.\ emeasure\ (Mi\ j)\ (Ai\ i)) = (\sum i{\in}I.\ emeasure\ (Mi\ i)\ (\bigcup (range\ Ai)))$
by(*auto simp*: *suminf-sum intro*!: *Finite-Cartesian-Product.sum-cong-aux suminf-emeasure h*)
qed
qed(*auto simp*: *positive-def sum-measure-def intro*!: *sets.sets-into-space*)

lemma *sum-measure-infinite*: *infinite* $I \Longrightarrow$ *sum-measure M I Mi* = *null-measure M*
by(*auto simp*: *sum-measure-def null-measure-def*)

lemma *nn-integral-sum-measure*:
assumes $f \in$ *borel-measurable M* **and** [*measurable-cong*]: $\bigwedge i.\ i \in I \Longrightarrow sets\ (Mi\ i) = sets\ M$
shows $(\int^+ x.\ f\ x\ \partial sum\text{-}measure\ M\ I\ Mi) = (\sum i{\in}I.\ (\int^+ x.\ f\ x\ \partial(Mi\ i)))$
using *assms*(*1*)
proof *induction*
case *h*:(*cong f g*)
then show *?case* (**is** *?lhs* = *?rhs*)
by(*auto cong*: *nn-integral-cong*[*of sum-measure M I Mi,simplified*] *intro*!: *Finite-Cartesian-Product.sum-cong-aux*)
(*auto cong*: *nn-integral-cong simp*: *sets-eq-imp-space-eq*[*OF assms*(*2*)[*symmetric*]])
next
case (*set A*)
then show *?case*
by(*auto simp*: *emeasure-sum-measure assms*)
next
case (*mult u c*)
then show *?case*
by(*auto simp add*: *nn-integral-cmult sum-distrib-left intro*!: *Finite-Cartesian-Product.sum-cong-aux*)
next
case (*add u v*)
then show *?case*
by(*auto simp*: *nn-integral-add sum.distrib*)
next
case *ih*[*measurable*]:(*seq U*)
show *?case* (**is** *?lhs* = *?rhs*)
proof −
have $?lhs = (\int^+ x.\ (\bigsqcup i.\ U\ i\ x)\ \partial sum\text{-}measure\ M\ I\ Mi)$
by(*auto intro*!: *nn-integral-cong*) (*use SUP-apply* **in** *auto*)
also have ... $= (\bigsqcup i.\ (\int^+ x.\ U\ i\ x\ \partial sum\text{-}measure\ M\ I\ Mi))$
by(*rule nn-integral-monotone-convergence-SUP*) (*use ih* **in** *auto*)
also have ... $= (\bigsqcup i.\ \sum j{\in}I.\ (\int^+ x.\ U\ i\ x\ \partial(Mi\ j)))$
by(*simp add*: *ih*)
also have ... $= (\sum j{\in}I.\ \bigsqcup i.\ \int^+ x.\ U\ i\ x\ \partial(Mi\ j))$
by(*auto intro*!: *incseq-nn-integral ih ennreal-SUP-sum*)
also have ... $= (\sum j{\in}I.\ \int^+ x.\ (\bigsqcup i.\ U\ i\ x)\ \partial(Mi\ j))$

```
    by(auto intro!: Finite-Cartesian-Product.sum-cong-aux nn-integral-monotone-convergence-SUP[symmetric]
ih)
    also have ... = ?rhs
    by(auto intro!: Finite-Cartesian-Product.sum-cong-aux nn-integral-cong) (metis
SUP-apply Sup-apply)
    finally show ?thesis .
  qed
qed

corollary integrable-sum-measure-iff-ne:
  fixes f :: 'a ⇒ 'b::{banach, second-countable-topology}
  assumes [measurable-cong]: ⋀i. i ∈ I ⟹ sets (Mi i) = sets M and finite I and
I ≠ {}
  shows integrable (sum-measure M I Mi) f ⟷ (∀ i∈I. integrable (Mi i) f)
proof safe
  fix i
  assume [measurable]: integrable (sum-measure M I Mi) f and i:i ∈ I
  then have [measurable]:⋀i. i ∈ I ⟹ f ∈ borel-measurable (Mi i)
   f ∈ borel-measurable M (∫⁺ x. ennreal (norm (f x)) ∂sum-measure M I Mi) <
∞
    by(auto simp: integrable-iff-bounded)
  hence (∑ i∈I. ∫⁺ x. ennreal (norm (f x)) ∂Mi i) < ∞
    by(simp add: nn-integral-sum-measure assms)
  thus integrable (Mi i) f
    by(auto simp: assms integrable-iff-bounded i)
next
  assume h:∀ i∈I. integrable (Mi i) f
  obtain i where i:i ∈ I
    using assms by auto
  have [measurable]: f ∈ borel-measurable M
    using h[rule-format,OF i] i by auto
  show integrable (sum-measure M I Mi) f
    using h by(auto simp: integrable-iff-bounded nn-integral-sum-measure assms)
qed

corollary integrable-sum-measure-iff:
  fixes f :: 'a ⇒ 'b::{banach, second-countable-topology}
  assumes [measurable-cong]: ⋀i. i ∈ I ⟹ sets (Mi i) = sets M and finite I
      and [measurable]: f ∈ borel-measurable M
    shows integrable (sum-measure M I Mi) f ⟷ (∀ i∈I. integrable (Mi i) f)
proof safe
  fix i
  assume integrable (sum-measure M I Mi) f i ∈ I
  thus integrable (Mi i) f
    using integrable-sum-measure-iff-ne[of I Mi,OF assms(1−2)] by auto
qed(auto simp: integrable-iff-bounded nn-integral-sum-measure assms)

lemma integral-sum-measure:
  fixes f :: 'a ⇒ 'b::{banach, second-countable-topology}
```

assumes [*measurable-cong*]:$\bigwedge i.\ i \in I \Longrightarrow$ *sets* (*Mi i*) = *sets M* $\bigwedge i.\ i \in I \Longrightarrow$ *integrable* (*Mi i*) *f*
shows ($\int x.\ f\ x\ \partial$*sum-measure M I Mi*) = ($\sum i{\in}I.$ ($\int x.\ f\ x\ \partial$(*Mi i*)))
proof −
consider $I = \{\}$ | *finite I* $I \neq \{\}$ | *infinite I* **by** *auto*
then show *?thesis*
proof *cases*
case *1*
then show *?thesis*
by(*auto simp*: *sum-measure-def integral-null-measure*[*simplified null-measure-def*])
next
case *2*
have *integrable* (*sum-measure M I Mi*) *f*
by(*auto simp*: *assms*(*2*) *integrable-sum-measure-iff-ne*[*of I Mi*,*OF assms*(*1*) *2*,*simplified*])
thus *?thesis*
proof *induction*
case *h*:(*base A c*)
then have h':$\bigwedge i.\ i \in I \Longrightarrow$ *emeasure* (*Mi i*) $A < \top$
by(*auto simp*: *emeasure-sum-measure assms 2*)
show *?case*
using *h*
by(*auto simp*: *measure-def* h' *emeasure-sum-measure assms enn2real-sum*[*of I* $\lambda i.$ *emeasure* (*Mi i*) *A*,*OF* h'] *scaleR-left.sum*
intro!: *Finite-Cartesian-Product.sum-cong-aux*)
next
case *ih*:(*add f g*)
then have $\bigwedge i.\ i \in I \Longrightarrow$ *integrable* (*Mi i*) *g* $\bigwedge i.\ i \in I \Longrightarrow$ *integrable* (*Mi i*) *f*
by(*auto simp*: *integrable-sum-measure-iff-ne assms 2*)
with *ih* **show** *?case*
by(*auto simp*: *sum.distrib*)
next
case *ih*:(*lim f s*)
then have [*measurable*]:$f \in$ *borel-measurable M* $\bigwedge i.\ s\ i \in$ *borel-measurable M*
by *auto*
have *int*[*measurable*]:*integrable* (*Mi i*) *f* $\bigwedge j.$ *integrable* (*Mi i*) (*s j*) **if** $i \in I$ **for** *i*
using *that ih 2* **by**(*auto simp add*: *integrable-sum-measure-iff assms*)
show *?case*
proof(*rule LIMSEQ-unique*[**where** $X{=}\lambda i.\ \sum j{\in}I.\ \int x.\ s\ i\ x\ \partial(Mi\ j)$])
show ($\lambda i.\ \sum j{\in}I.\ \int x.\ s\ i\ x\ \partial(Mi\ j)$) $\longrightarrow$ ($\int x.\ f\ x\ \partial$*sum-measure M I Mi*)
using *ih* **by**(*auto simp*: *ih*(*5*)[*symmetric*] *intro*!: *integral-dominated-convergence*[**where** $w{=}\lambda x.\ 2{*}norm\ (f\ x)$])
show ($\lambda i.\ \sum j{\in}I.\ \int x.\ s\ i\ x\ \partial(Mi\ j)$) $\longrightarrow$ ($\sum j{\in}I.$ ($\int x.\ f\ x\ \partial(Mi\ j)$))
proof(*rule tendsto-sum*)
fix *j*
assume *j*: $j \in I$
show ($\lambda i.\ \int x.\ s\ i\ x\ \partial(Mi\ j)$) $\longrightarrow$ ($\int x.\ f\ x\ \partial(Mi\ j)$)

using *integral-dominated-convergence*[*of f Mi j s* $\lambda x.\ 2 * norm\ (f\ x)$,*OF*
- - - *AE-I2 AE-I2*] *ih int*[*OF j*]
by(*auto simp*: *sets-eq-imp-space-eq*[*OF assms*(*1*)[*OF j*]])
qed
qed
qed
next
case *3*
then show *?thesis*
by(*simp add*: *sum-measure-infinite*)
qed
qed

Lemmas related to scale measure

lemma *integrable-scale-measure*:
fixes $f :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$
assumes *integrable M f*
shows *integrable* (*scale-measure* (*ennreal r*) *M*) *f*
using *assms ennreal-less-top*
by(*auto simp*: *integrable-iff-bounded nn-integral-scale-measure ennreal-mult-less-top*)

lemma *integral-scale-measure*:
assumes $r \geq 0$ *integrable M f*
shows $(\int x.\ f\ x\ \partial scale\text{-}measure\ (ennreal\ r)\ M) = r * (\int x.\ f\ x\ \partial M)$
using *assms*(*2*)
proof *induction*
case *ih*:(*lim f s*)
show *?case*
proof(*rule LIMSEQ-unique*[**where** $X = \lambda i.\ \int x.\ s\ i\ x\ \partial scale\text{-}measure\ (ennreal\ r)$
M])
from *ih*(*1*−*4*) **show** $(\lambda i.\ \int x.\ s\ i\ x\ \partial scale\text{-}measure\ (ennreal\ r)\ M) \longrightarrow (\int x.$
$f\ x\ \partial scale\text{-}measure\ (ennreal\ r)\ M)$
by(*auto intro*!: *integral-dominated-convergence*[**where** $w = \lambda x.\ 2 * norm\ (f\ x)$]
integrable-scale-measure
simp: *space-scale-measure*)
show $(\lambda i.\ \int x.\ s\ i\ x\ \partial scale\text{-}measure\ (ennreal\ r)\ M) \longrightarrow r * (\int x.\ f\ x\ \partial M)$
unfolding *ih*(*5*) **using** *ih*(*1*−*4*) **by**(*auto intro*!: *integral-dominated-convergence*[**where**
$w = \lambda x.\ 2 * norm\ (f\ x)$] *tendsto-mult-left*)
qed
qed(*auto simp*: *measure-scale-measure*[*OF assms*(*1*)] *ring-class.ring-distribs*(*1*) *integrable-scale-measure*)

lemma
fixes *c* :: *ereal*
assumes *c*: $c \neq -\infty$ **and** *a*: $\bigwedge n.\ 0 \leq a\ n$
shows *liminf-cadd*: $liminf\ (\lambda n.\ c + a\ n) = c + liminf\ a$
and *limsup-cadd*: $limsup\ (\lambda n.\ c + a\ n) = c + limsup\ a$
by(*auto simp add*: *liminf-SUP-INF limsup-INF-SUP INF-ereal-add-right*[*OF - c*
a] *SUP-ereal-add-right*[*OF - c*]

```
    intro!: INF-ereal-add-right c SUP-upper2 a)

lemma(in Metric-space) frontier-measure-zero-balls:
  assumes sets N = sets (borel-of mtopology) finite-measure N M ≠ {}
    and e > 0 and separable-space mtopology
  obtains ai ri where
    (⋃i::nat. mball (ai i) (ri i)) = M (⋃i::nat. mcball (ai i) (ri i)) = M
    ⋀i. ai i ∈ M ⋀i. ri i > 0 ⋀i. ri i < e
    ⋀i. measure N (mtopology frontier-of (mball (ai i) (ri i))) = 0
    ⋀i. measure N (mtopology frontier-of (mcball (ai i) (ri i))) = 0
proof −
  interpret N: finite-measure N by fact
  have [measurable]: ⋀a r. mball a r ∈ sets N ⋀a r. mcball a r ∈ sets N
    ⋀a r. mtopology frontier-of (mball a r) ∈ sets N ⋀a r. mtopology frontier-of
(mcball a r) ∈ sets N
   by(auto simp: assms(1) borel-of-closed borel-of-open[OF openin-mball] closedin-frontier-of)
  have mono:mtopology frontier-of (mball a r) ⊆ {y∈M. d a y = r}
    mtopology frontier-of (mcball a r) ⊆ {y∈M. d a y = r} for a r
  proof −
    have mtopology frontier-of (mball a r) ⊆ mcball a r − mball a r
       using closure-of-mball by(auto simp: frontier-of-def interior-of-openin[OF
openin-mball])
    also have ... ⊆ {y∈M. d a y = r}
      by auto
    finally show mtopology frontier-of (mball a r) ⊆ {y∈M. d a y = r} .
    have mtopology frontier-of (mcball a r) ⊆ mcball a r − mball a r
      using interior-of-mcball by(auto simp: frontier-of-def closure-of-closedin[OF
closedin-mcball])
    also have ... ⊆ {y∈M. d a y = r}
      by(auto simp: mcball-def mball-def)
    finally show mtopology frontier-of (mcball a r) ⊆ {y∈M. d a y = r} .
  qed
  have sets[measurable]: {y∈M. d a y = r} ∈ sets N if a ∈ M for a r
  proof −
    have [simp]: d a −‘ {r} ∩ M = {y∈M. d a y = r} by blast
    show ?thesis
     using measurable-sets[OF continuous-map-measurable[OF uniformly-continuous-imp-continuous-map[OF
mdist-set-uniformly-continuous[of Self {a}]]],of {r}]
        by(simp add: borel-of-euclidean mtopology-of-def space-borel-of assms(1)
mdist-set-Self)
       (metis (no-types, lifting) ‹d a −‘ {r} ∩ M = {y ∈ M. d a y = r}› commute
d-set-singleton that vimage-inter-cong)
  qed
  from assms(5) obtain U where U: countable U mdense U by(auto simp: sep-
arable-space-def2)
  with assms(3) have U-ne: U ≠ {}
    by(auto simp: mdense-empty-iff)
  { fix i :: nat
    have countable {r ∈ {e/2<..<e}. measure N {y ∈ M. d (from-nat-into U i) y
```

= *r*} ≠ *0*}
by(*rule N.countable-support-sets*) (*auto simp*: *disjoint-family-on-def*)
from *real-interval-avoid-countable-set*[*of e / 2 e*,*OF - this*] *assms*(*4*)
have ∃ *r*. *measure N* {*y*∈*M*. *d* (*from-nat-into U i*) *y* = *r*} = *0* ∧ *r* > *e* / *2* ∧ *r* < *e*
by *auto*
}
then obtain *ri* **where** *ri*: ⋀*i*. *measure N* {*y*∈*M*. *d* (*from-nat-into U i*) *y* = *ri i*} = *0*
⋀*i*. *ri i* > *e* / *2* ⋀*i*. *ri i* < *e*
by *metis*
have *1*: (⋃*i*. *mball* (*from-nat-into U i*) (*ri i*)) = *M* (⋃*i*. *mcball* (*from-nat-into U i*) (*ri i*)) = *M*
proof −
have *M* = (⋃*u*∈*U*. *mball u* (*e* / *2*))
by(*rule mdense-balls-cover*[*OF U*(*2*),*symmetric*]) (*simp add*: *assms*(*4*))
also have ... = (⋃*i*. *mball* (*from-nat-into U i*) (*e* / *2*))
by(*rule UN-from-nat-into*[*OF U*(*1*) *U-ne*])
also have ... ⊆ (⋃*i*. *mball* (*from-nat-into U i*) (*ri i*))
using *mball-subset-concentric*[*OF order.strict-implies-order*[*OF ri*(*2*)]] **by** *auto*
finally have *1*:*M* ⊆ (⋃*i*. *mball* (*from-nat-into U i*) (*ri i*)) **.**
moreover have *M* ⊆ (⋃*i*. *mcball* (*from-nat-into U i*) (*ri i*))
by(*rule order.trans*[*OF 1*]) *fastforce*
ultimately show (⋃*i*. *mball* (*from-nat-into U i*) (*ri i*)) = *M* (⋃*i*. *mcball* (*from-nat-into U i*) (*ri i*)) = *M*
by *fastforce*+
qed
have *2*: ⋀*i*. *from-nat-into U i* ∈ *M* ⋀*i*. *ri i* > *0* ⋀*i*. *ri i* < *e*
using *from-nat-into*[*OF U-ne*] *dense-in-subset*[*OF U*(*2*)] *ri*(*3*) *assms*(*4*)
by(*auto intro*!: *order.strict-trans*[*OF - ri*(*2*),*of 0*])
have *3*: *measure N* (*mtopology frontier-of* (*mball* (*from-nat-into U i*) (*ri i*))) = *0*
measure N (*mtopology frontier-of* (*mcball* (*from-nat-into U i*) (*ri i*))) = *0* **for** *i*
using *N.finite-measure-mono*[*OF mono*(*1*) *sets*[*of from-nat-into U i ri i*]]
N.finite-measure-mono[*OF mono*(*2*) *sets*[*of from-nat-into U i ri i*]]
by (*auto simp add*: *2 measure-le-0-iff ri*(*1*))
show *?thesis*
using *1 2 3 that* **by** *blast*
qed

lemma *finite-measure-integral-eq-dense*:
assumes *finite*: *finite-measure N finite-measure M*
and *sets-N*:*sets N* = *sets* (*borel-of X*) **and** *sets-M*: *sets M* = *sets* (*borel-of X*)
and *dense*:*dense-in* (*mtopology-of* (*cfunspace X euclidean-metric*)) *F*
and *integ-eq*:⋀*f*::- ⇒ *real*. *f* ∈ *F* ⟹ (∫ *x*. *f x* ∂*N*) = (∫ *x*. *f x* ∂*M*)
and *f*:*continuous-map X euclideanreal f bounded* (*f* ‘ *topspace X*)
shows (∫ *x*. *f x* ∂*N*) = (∫ *x*. *f x* ∂*M*)
proof −

interpret *N*: *finite-measure N*
by *fact*
interpret *M*: *finite-measure M*
by *fact*
have *integ-N*: $\bigwedge A.\ A \in sets\ N \Longrightarrow integrable\ N\ (indicat\text{-}real\ A)$
and *integ-M*: $\bigwedge A.\ A \in sets\ M \Longrightarrow integrable\ M\ (indicat\text{-}real\ A)$
by (*auto simp add*: *N.emeasure-eq-measure M.emeasure-eq-measure*)
have *space-N*: *space N* = *topspace X* **and** *space-M*: *space M* = *topspace X*
using *sets-N sets-M sets-eq-imp-space-eq*[*of - borel-of X*]
by(*auto simp*: *space-borel-of*)
from *f* **obtain** *B* **where** *B*: $\bigwedge x.\ x \in topspace\ X \Longrightarrow |f\ x| \le B$
by (*meson bounded-real imageI*)
show $(\int x.\ f\ x\ \partial N) = (\int x.\ f\ x\ \partial M)$
proof −
have *in-mspace-measurable*: *g* ∈ *borel-measurable N g* ∈ *borel-measurable M*
if *g*:*g* ∈ *mspace* (*cfunspace X* (*euclidean-metric* :: *real metric*)) **for** *g*
using *continuous-map-measurable*[*of X euclidean*,*simplified borel-of-euclidean*] *g*
by(*auto simp*: *sets-M cong*: *measurable-cong-sets sets-N*)
have *f'*:(λ*x*∈*topspace X*. *f x*) ∈ *mspace* (*cfunspace X euclidean-metric*)
using *f*(*1*) *f*(*2*) **by** *simp*
with *mdense-of-def3*[*THEN iffD1*,*OF assms*(*5*)] **obtain** *fn* **where** *fn*:
range fn ⊆ *F limitin* (*mtopology-of* (*cfunspace X euclidean-metric*)) *fn* (λ*x*∈*topspace X*. *f x*) *sequentially*
by *blast*
hence *fn-space*: $\bigwedge n.$ *fn n* ∈ *mspace* (*cfunspace X euclidean-metric*)
using *dense-in-subset*[*OF assms*(*5*)] **by** *auto*
hence [*measurable*]: (λ*x*∈*topspace X*. *f x*) ∈ *borel-measurable N* (λ*x*∈*topspace X*. *f x*) ∈ *borel-measurable M*
$\bigwedge n.$ *fn n* ∈ *borel-measurable N* $\bigwedge n.$ *fn n* ∈ *borel-measurable M*
using *f'* **by** (*auto simp del*: *mspace-cfunspace intro*!: *in-mspace-measurable*)
interpret *d*: *Metric-space mspace* (*cfunspace X euclidean-metric*) *mdist* (*cfunspace X* (*euclidean-metric* :: *real metric*))
by *blast*
from *fn* **have** *limitin d.mtopology fn* (λ*x*∈*topspace X*. *f x*) *sequentially*
by (*simp add*: *mtopology-of-def*)
hence *limit*:$\bigwedge \varepsilon.\ \varepsilon > 0 \Longrightarrow \exists N.\ \forall n \ge N.$ *fn n* ∈ *mspace* (*cfunspace X euclidean-metric*) ∧
mdist (*cfunspace X euclidean-metric*) (*fn n*) (*restrict f* (*topspace X*)) < ε
unfolding *d.limit-metric-sequentially* **by** *blast*
from *this*[*of 1*] **obtain** *N0* **where** *N0*:
$\bigwedge n.\ n \ge N0 \Longrightarrow$ *mdist* (*cfunspace X euclidean-metric*) (*fn n*) (λ*x*∈*topspace X*. *f x*) < *1*
by *auto*
have *1*: (λ*i*. *fn* (*i* + *N0*) *x*) ⟶ (λ*x*∈*topspace X*. *f x*) *x* **if** *x*:*x* ∈ *topspace X* **for** *x*
proof(*rule LIMSEQ-I*)
fix *r* :: *real*

```
    assume r:0 < r
    from limit[OF half-gt-zero[OF r]] obtain N where N:
      ⋀n. n ≥ N ⟹ mdist (cfunspace X euclidean-metric) (fn n) (restrict f
(topspace X)) < r / 2
      by blast
    show ∃no. ∀n≥no. norm (fn (n + N0) x − restrict f (topspace X) x) < r
    proof(safe intro!: exI[where x=N])
      fix n
      assume n:N ≤ n
      with N[OF trans-le-add1[OF this,of N0]]
     have mdist (cfunspace X euclidean-metric) (fn (n + N0)) (restrict f (topspace
X)) ≤ r / 2
        by auto
       from order.strict-trans1[OF mdist-cfunspace-imp-mdist-le[OF fn-space f′
this x],of r] x r
      show norm (fn (n + N0) x − restrict f (topspace X) x) < r
        by (auto simp: dist-real-def)
    qed
  qed
  have 2: norm (fn (i + N0) x) ≤ 2 ∗ B + 1 if x:x ∈ topspace X for i x
  proof−
    from N0[of i + N0]
   have mdist (cfunspace X euclidean-metric) (fn (i + N0)) (restrict f (topspace
X)) ≤ 1
      by linarith
    from mdist-cfunspace-imp-mdist-le[OF fn-space f′ this x]
    have norm (fn (i + N0) x − f x) ≤ 1
      using x by (auto simp: dist-real-def)
    thus ?thesis
      using B[OF x] by auto
  qed
  from 1 2 have (λi. integral^L N (fn (i + N0))) ⟶ integral^L N (restrict f
(topspace X))
   by(auto intro!: integral-dominated-convergence[where s=λi. fn (i + N0) and
w=λx. 2 ∗ B + 1]
           simp: space-N)
  moreover have (λi. integral^L N (fn (i + N0))) ⟶ integral^L M (restrict
f (topspace X))
  proof −
    have [simp]:integral^L N (fn (i + N0)) = integral^L M (fn (i + N0)) for i
      using fn(1) by(auto intro!: assms(6))
    from 1 2 show ?thesis
       by(auto intro!: integral-dominated-convergence[where s=λi. fn (i + N0)
and w=λx. 2 ∗ B + 1]
           simp: space-M)
  qed
  ultimately have integral^L N (restrict f (topspace X)) = integral^L M (restrict
f (topspace X))
    by(rule tendsto-unique[OF sequentially-bot])
```

moreover have *integral*^{L} *N* (*restrict f* (*topspace X*)) = *integral*^{L} *N f*
by(*auto cong*: *Bochner-Integration.integral-cong*[*OF refl*] *simp*: *space-N*[*symmetric*])
moreover have *integral*^{L} *M* (*restrict f* (*topspace X*)) = *integral*^{L} *M f*
by(*auto cong*: *Bochner-Integration.integral-cong*[*OF refl*] *simp*: *space-M*[*symmetric*])
ultimately show *?thesis*
by *simp*
qed
qed

1.2 Sequentially Continuous Maps

definition *seq-continuous-map* :: $'a$ *topology* ⇒ $'b$ *topology* ⇒ ($'a$ ⇒ $'b$) ⇒ *bool*
where
seq-continuous-map X Y f ≡ (∀ *xn x*. *limitin X xn x sequentially* ⟶ *limitin Y* (λ*n*. *f* (*xn n*)) (*f x*) *sequentially*)

lemma *seq-continuous-map*:
seq-continuous-map X Y f ⟷ (∀ *xn x*. *limitin X xn x sequentially* ⟶ *limitin Y* (λ*n*. *f* (*xn n*)) (*f x*) *sequentially*)
by(*auto simp*: *seq-continuous-map-def*)

lemma *seq-continuous-map-funspace*:
assumes *seq-continuous-map X Y f*
shows *f* ∈ *topspace X* → *topspace Y*
proof
fix *x*
assume *x* ∈ *topspace X*
then have *limitin X* (λ*n*. *x*) *x sequentially*
by *auto*
hence *limitin Y* (λ*n*. *f x*) (*f x*) *sequentially*
using *assms*
by (*meson limitin-sequentially seq-continuous-map*)
thus *f x* ∈ *topspace Y*
by *auto*
qed

lemma *seq-continuous-iff-continuous-first-countable*:
assumes *first-countable X*
shows *seq-continuous-map X Y* = *continuous-map X Y*
by *standard* (*simp add*: *continuous-map-iff-limit-seq assms seq-continuous-map-def*)

1.3 Sequential Compactness

definition *seq-compactin* :: $'a$ *topology* ⇒ $'a$ *set* ⇒ *bool* **where**
seq-compactin X S
⟷ *S* ⊆ *topspace X* ∧ (∀ *xn*. (∀ *n*::*nat*. *xn n* ∈ *S*) ⟶ (∃ *l*∈*S*. ∃ *a*::*nat* ⇒ *nat*. *strict-mono a* ∧ *limitin X* (*xn* ∘ *a*) *l sequentially*))

definition *seq-compact-space X* ≡ *seq-compactin X* (*topspace X*)

lemma *seq-compactin-subset-topspace*: *seq-compactin X S* ⟹ *S* ⊆ *topspace X*
by(*auto simp*: *seq-compactin-def*)

lemma *seq-compactin-empty*[*simp*]: *seq-compactin X* {}
by(*auto simp*: *seq-compactin-def*)

lemma *seq-compactin-seq-compact*[*simp*]: *seq-compactin euclidean S* ⟷ *seq-compact S*
by(*auto simp*: *seq-compactin-def seq-compact-def*)

lemma *image-seq-compactin*:
assumes *seq-compactin X S seq-continuous-map X Y f*
shows *seq-compactin Y* (*f* ' *S*)
unfolding *seq-compactin-def*
proof *safe*
fix *yn*
assume ∀ *n*::*nat*. *yn n* ∈ *f* ' *S*
then have ∀ *n*. ∃ *x*∈*S*. *yn n* = *f x*
by *blast*
then obtain *xn* **where** *xn*: ⋀*n*::*nat*. *xn n* ∈ *S* ⋀*n*. *yn n* = *f* (*xn n*)
by *metis*
then obtain *lx a* **where** *la*: *lx* ∈ *S strict-mono a limitin X* (*xn* ∘ *a*) *lx sequentially*
by (*meson assms*(*1*) *seq-compactin-def*)
show ∃ *l*∈*f* ' *S*. ∃ *a*. *strict-mono a* ∧ *limitin Y* (*yn* ∘ *a*) *l sequentially*
proof(*safe intro*!: *bexI*[**where** *x*=*f lx*] *exI*[**where** *x*=*a*])
have [*simp*]:*yn* ∘ *a* = (λ*n*. *f* ((*xn* ∘ *a*) *n*))
by(*auto simp*: *xn*(*2*) *comp-def*)
show *limitin Y* (*yn* ∘ *a*) (*f lx*) *sequentially*
using *la*(*3*) *assms*(*2*) *xn*(*1*,*2*) **by**(*fastforce simp*: *seq-continuous-map*)
qed(*use la* **in** *auto*)
qed(*use seq-compactin-subset-topspace*[*OF assms*(*1*)] *seq-continuous-map-funspace*[*OF assms*(*2*)] **in** *auto*)

lemma *closed-seq-compactin*:
assumes *seq-compactin X K C* ⊆ *K closedin X C*
shows *seq-compactin X C*
unfolding *seq-compactin-def*
proof *safe*
fix *xn*
assume *xn*: ∀ *n*::*nat*. *xn n* ∈ *C*
then have ∀ *n*. *xn n* ∈ *K*
using *assms*(*2*) **by** *blast*
with *assms*(*1*) **obtain** *l a* **where** *l*: *l* ∈ *K strict-mono a limitin X* (*xn* ∘ *a*) *l sequentially*
by (*meson seq-compactin-def*)
have *l* ∈ *C*
using *xn* **by**(*auto intro*!: *limitin-closedin*[*OF l*(*3*) *assms*(*3*)])
with *l*(*2*,*3*) **show** ∃ *l*∈*C*. ∃ *a*. *strict-mono a* ∧ *limitin X* (*xn* ∘ *a*) *l sequentially*
by *blast*

qed(*use closedin-subset*[*OF assms*(*3*)] **in** *auto*)

corollary *closedin-seq-compact-space*:
seq-compact-space X ⟹ *closedin X C* ⟹ *seq-compactin X C*
by(*auto intro*!: *closed-seq-compactin*[**where** *K*=*topspace X* **and** *C*=*C*] *closedin-subset simp*: *seq-compact-space-def*)

lemma *seq-compactin-subtopology*: *seq-compactin* (*subtopology X S*) *T* ⟷ *seq-compactin X T* ∧ *T* ⊆ *S*
by(*fastforce simp*: *seq-compactin-def limitin-subtopology subsetD*)

corollary *seq-compact-space-subtopology*: *seq-compactin X S* ⟹ *seq-compact-space* (*subtopology X S*)
by(*auto simp*: *seq-compact-space-def seq-compactin-subtopology inf-absorb2 seq-compactin-subset-topspace*)

lemma *seq-compactin-PiED*:
assumes *seq-compactin* (*product-topology X I*) (Pi_E *I S*)
shows (Pi_E *I S* = {} ∨ (∀ *i*∈*I*. *seq-compactin* (*X i*) (*S i*)))
proof −
consider Pi_E *I S* = {} | Pi_E *I S* ≠ {}
by *blast*
then show (Pi_E *I S* = {} ∨ (∀ *i*∈*I*. *seq-compactin* (*X i*) (*S i*)))
proof *cases*
case *1*
then show *?thesis*
by *simp*
next
case *2*
then have *Si-ne*:⋀*i*. *i* ∈ *I* ⟹ *S i* ≠ {}
by *blast*
then obtain *ci* **where** *ci*: ⋀*i*. *i* ∈ *I* ⟹ *ci i* ∈ *S i*
by (*meson PiE-E ex-in-conv*)
show *?thesis*
proof(*safe intro*!: *disjI2*)
fix *i*
assume *i*: *i* ∈ *I*
show *seq-compactin* (*X i*) (*S i*)
unfolding *seq-compactin-def*
proof *safe*
fix *xn*
assume *xn*:∀ *n*::*nat*. *xn n* ∈ *S i*
define *Xn* **where** *Xn* ≡ (λ*n*. λ*j*∈*I*. *if j* = *i then xn n else ci j*)
have ⋀*n*. *Xn n* ∈ Pi_E *I S*
using *i xn ci* **by**(*auto simp*: *Xn-def*)
then obtain *L a* **where** *L*: *L* ∈ Pi_E *I S strict-mono a*
limitin (*product-topology X I*) (*Xn* ∘ *a*) *L sequentially*
by (*meson assms seq-compactin-def*)
thus ∃ *l*∈*S i*. ∃ *a*. *strict-mono a* ∧ *limitin* (*X i*) (*xn* ∘ *a*) *l sequentially*
using *i* **by**(*auto simp*: *limitin-componentwise Xn-def comp-def intro*!:

bexI[**where** *x=L i*] *exI*[**where** *x=a*])
next
show ⋀*x*. *x* ∈ *S i* ⟹ *x* ∈ *topspace* (*X i*)
using *i subset-PiE*[*THEN iffD1*,*OF seq-compactin-subset-topspace*[*OF assms*,*simplified*]] *2* **by** *auto*
qed
qed
qed
qed

lemma *metrizable-seq-compactin-iff-compactin*:
assumes *metrizable-space X*
shows *seq-compactin X S* ⟷ *compactin X S*
proof −
obtain *d* **where** *d*: *Metric-space* (*topspace X*) *d Metric-space.mtopology* (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms metrizable-space-def*)
interpret *Metric-space topspace X d*
by *fact*
have *seq-compactin X S* ⟷ *seq-compactin mtopology S*
by(*simp add*: *d*)
also have ... ⟷ *compactin mtopology S*
by(*fastforce simp*: *compactin-sequentially seq-compactin-def*)
also have ... ⟷ *compactin X S*
by(*simp add*: *d*)
finally show *?thesis* **.**
qed

corollary *metrizable-seq-compact-space-iff-compact-space*:
shows *metrizable-space X* ⟹ *seq-compact-space X* ⟷ *compact-space X*
unfolding *seq-compact-space-def compact-space-def* **by**(*rule metrizable-seq-compactin-iff-compactin*)

1.4 Lemmas for Limsup and Liminf

lemma *real-less-add-ex-less-pair*:
fixes *x w v* :: *real*
assumes $x < w + v$
shows ∃ *y z*. $x = y + z \wedge y < w \wedge z < v$
apply(*rule exI*[**where** $x = w - (w + v - x) / 2$])
apply(*rule exI*[**where** $x = v - (w + v - x) / 2$])
using *assms* **by** *auto*

lemma *ereal-less-add-ex-less-pair*:
fixes *x w v* :: *ereal*
assumes $-\infty < w$ $-\infty < v$ $x < w + v$
shows ∃ *y z*. $x = y + z \wedge y < w \wedge z < v$
proof −
consider $x = -\infty$ | $-\infty < x$ $x < \infty$ $w = \infty$ $v = \infty$
| $-\infty < x$ $x < \infty$ $w < \infty$ $v = \infty$ | $-\infty < x$ $x < \infty$ $v < \infty$ $w = \infty$

```
  | − ∞ < x x < ∞ w < ∞ v < ∞
  using assms(3) less-ereal.simps(2) by blast
 then show ?thesis
 proof cases
  assume x = − ∞
  then show ?thesis
   using assms by(auto intro!: exI[where x=− ∞])
 next
  assume h:− ∞ < x x < ∞ w = ∞ v = ∞
  show ?thesis
   apply(rule exI[where x=0])
   apply(rule exI[where x=x])
   using h assms by simp
 next
  assume h:− ∞ < x x < ∞ w < ∞ v = ∞
  then obtain x′ w′ where eq: w = ereal w′ x = ereal x′
   using assms by (metis less-irrefl sgn-ereal.cases)
  show ?thesis
   apply(rule exI[where x=w − 1])
   apply(rule exI[where x=x − (w − 1)])
   using h assms by(auto simp: eq one-ereal-def)
 next
  assume h:− ∞ < x x < ∞ v < ∞ w = ∞
  then obtain x′ v′ where eq: v = ereal v′ x = ereal x′
   using assms by (metis less-irrefl sgn-ereal.cases)
  show ?thesis
   apply(rule exI[where x=x − (v − 1)])
   apply(rule exI[where x=v − 1])
   using h assms by(auto simp: eq one-ereal-def)
 next
  assume − ∞ < x x < ∞ w < ∞ v < ∞
  then obtain x′ v′ w′ where eq: x = ereal x′ w = ereal w′ v = ereal v′
   using assms by (metis less-irrefl sgn-ereal.cases)
  have ∃ y′ z′. x′ = y′ + z′ ∧ y′ < w′ ∧ z′ < v′
   using real-less-add-ex-less-pair assms by(simp add: eq)
  then obtain y′ z′ where yz′:x′ = y′ + z′ ∧ y′ < w′ ∧ z′ < v′
   by blast
  show ?thesis
   apply(rule exI[where x=ereal y′])
   apply(rule exI[where x=ereal z′])
   using yz′ by(simp add: eq)
 qed
qed

lemma real-add-less:
 fixes x w v :: real
 assumes w + v < x
 shows ∃ y z. x = y + z ∧ w < y ∧ v < z
 apply(rule exI[where x=w + (x − (w + v)) / 2])
```

```
  apply(rule exI[where x=v + (x − (w + v)) / 2])
  using assms by auto

lemma ereal-add-less:
  fixes x w v :: ereal
  assumes w + v < x
  shows ∃ y z. x = y + z ∧ w < y ∧ v < z
proof −
  have − ∞ < x v < ∞ w < ∞
    using assms less-ereal.simps(2,3) by auto
  then consider x = ∞ w < ∞ v < ∞ | − ∞ < x x < ∞ w = − ∞ v = − ∞
    | − ∞ < x x < ∞ w = − ∞ v < ∞ − ∞ < v
    | − ∞ < x x < ∞ v = − ∞ w < ∞ − ∞ < w
    | − ∞ < x x < ∞ − ∞ < w w < ∞ v < ∞ − ∞ < v
    by blast
  thus ?thesis
  proof cases
    assume x = ∞ w < ∞ v < ∞
    then show ?thesis
      by(auto intro!: exI[where x=∞])
  next
    assume h:− ∞ < x x < ∞ w = − ∞ v = − ∞
    show ?thesis
      apply(rule exI[where x=0])
      apply(rule exI[where x=x])
      using h assms by simp
  next
    assume h:− ∞ < x x < ∞ w = − ∞ v < ∞ − ∞ < v
    then obtain x′ v′ where xv′: x = ereal x′ v = ereal v′
      by (metis less-irrefl sgn-ereal.cases)
    show ?thesis
      apply(rule exI[where x=x − (v + 1)])
      apply(rule exI[where x=v + 1])
      using h by(auto simp: xv′)
  next
    assume h:− ∞ < x x < ∞ v = − ∞ w < ∞ − ∞ < w
    then obtain x′ w′ where xw′: x = ereal x′ w = ereal w′
      by (metis less-irrefl sgn-ereal.cases)
    show ?thesis
      apply(rule exI[where x=w + 1])
      apply(rule exI[where x=x − (w + 1)])
      using h by(auto simp: xw′)
  next
    assume h:− ∞ < x x < ∞ − ∞ < w w < ∞ v < ∞ − ∞ < v
    then obtain x′ v′ w′ where eq: x = ereal x′ w = ereal w′ v = ereal v′
      using assms by (metis less-irrefl sgn-ereal.cases)
    have ∃ y′ z′. x′ = y′ + z′ ∧ y′ > w′ ∧ z′ > v′
      using assms real-add-less by(auto simp: eq)
    then obtain y′ z′ where yz′:x′ = y′ + z′ ∧ y′ > w′ ∧ z′ > v′
```

by *blast*
show *?thesis*
apply(*rule exI*[**where** *x*=*ereal y'*])
apply(*rule exI*[**where** *x*=*ereal z'*])
using *yz'* **by**(*simp add*: *eq*)
qed
qed

A generalized version of ¬ (*liminf ?u* = ∞ ∧ *liminf ?v* = − ∞ ∨ *liminf ?u* = − ∞ ∧ *liminf ?v* = ∞) ⟹ *liminf ?u* + *liminf ?v* ≤ *liminf* (λ*n*. *?u n* + *?v n*).

lemma *ereal-Liminf-add-mono*:
fixes *u v*::*'a* ⇒ *ereal*
assumes ¬((*Liminf F u* = ∞ ∧ *Liminf F v* = −∞) ∨ (*Liminf F u* = −∞ ∧ *Liminf F v* = ∞))
shows *Liminf F* (λ*n*. *u n* + *v n*) ≥ *Liminf F u* + *Liminf F v*
proof (*cases*)
assume (*Liminf F u* = −∞) ∨ (*Liminf F v* = −∞)
then have ∗: *Liminf F u* + *Liminf F v* = −∞ **using** *assms* **by** *auto*
show *?thesis* **by** (*simp add*: ∗)
next
assume ¬((*Liminf F u* = −∞) ∨ (*Liminf F v* = −∞))
then have *h*:*Liminf F u* > −∞ *Liminf F v* > −∞ **by** *auto*
show *?thesis*
unfolding *le-Liminf-iff*
proof *safe*
fix *y*
assume *y*: *y* < *Liminf F u* + *Liminf F v*
then obtain *x z* **where** *xz*: *y* = *x* + *z x* < *Liminf F u* *z* < *Liminf F v*
using *ereal-less-add-ex-less-pair h* **by** *blast*
show ∀$_F$ *x in F*. *y* < *u x* + *v x*
by(*rule eventually-mp*[*OF* - *eventually-conj*[*OF less-LiminfD*[*OF xz*(*2*)] *less-LiminfD*[*OF xz*(*3*)]]])
(*auto simp*: *xz intro*!: *eventuallyI ereal-add-strict-mono2*)
qed
qed

A generalized version of *limsup* (λ*n*. *?u n* + *?v n*) ≤ *limsup ?u* + *limsup ?v*.

lemma *ereal-Limsup-add-mono*:
fixes *u v*::*'a* ⇒ *ereal*
shows *Limsup F* (λ*n*. *u n* + *v n*) ≤ *Limsup F u* + *Limsup F v*
unfolding *Limsup-le-iff*
proof *safe*
fix *y*
assume *Limsup F u* + *Limsup F v* < *y*
then obtain *x z* **where** *xz*: *y* = *x* + *z Limsup F u* < *x Limsup F v* < *z*
using *ereal-add-less* **by** *blast*
show ∀$_F$ *x in F*. *u x* + *v x* < *y*

by(*rule eventually-mp*[*OF* - *eventually-conj*[*OF Limsup-lessD*[*OF xz*(*2*)] *Limsup-lessD*[*OF xz*(*3*)]]])
(*auto simp*: *xz intro*!: *eventuallyI ereal-add-strict-mono2*)
qed

1.5 A Characterization of Closed Sets by Limit

There is a net which charactrize closed sets in terms of convergence. Since Isabelle/HOL's convergent is defined through filters, we transform the net to a filter. We refer to the lecture notes by Shi [3] for the conversion.

definition *derived-filter* :: [$'i$ *set*, $'i \Rightarrow 'i \Rightarrow$ *bool*] $\Rightarrow$ $'i$ *filter* **where**
derived-filter I op $\equiv$ ($\bigsqcap$ *i*∈*I*. *principal* {*j*∈*I*. *op i j*})

lemma *eventually-derived-filter*:
assumes $A \neq \{\}$
and *refl*:$\bigwedge a.\ a \in A \Longrightarrow$ *rel a a*
and *trans*:$\bigwedge a\ b\ c.\ a \in A \Longrightarrow b \in A \Longrightarrow c \in A \Longrightarrow$ *rel a b* $\Longrightarrow$ *rel b c* $\Longrightarrow$ *rel a c*
and *pair-bounded*:$\bigwedge a\ b.\ a \in A \Longrightarrow b \in A \Longrightarrow \exists c{\in}A.$ *rel a c* $\wedge$ *rel b c*
shows *eventually P* (*derived-filter A rel*) $\longleftrightarrow$ ($\exists i{\in}A.\ \forall n{\in}A.$ *rel i n* $\longrightarrow$ *P n*)
proof −
show *?thesis*
unfolding *derived-filter-def*
proof(*subst eventually-INF-base*)
fix *a b*
assume *h*:$a \in A$ $b \in A$
then obtain *z* **where** $z \in A$ *rel a z rel b z*
using *pair-bounded* **by** *metis*
thus $\exists x{\in}A.$ *principal* $\{j \in A.$ *rel x j*$\}$ $\leq$ *principal* $\{j \in A.$ *rel a j*$\}$ $\sqcap$ *principal* $\{j \in A.$ *rel b j*$\}$
using *h* **by**(*auto intro*!: *bexI*[**where** *x*=*z*] *dest*: *trans*)
next
show ($\exists b{\in}A.$ *eventually P* (*principal* $\{j \in A.$ *rel b j*$\}$)) $\longleftrightarrow$ ($\exists i{\in}A.\ \forall n{\in}A.$ *rel i n* $\longrightarrow$ *P n*)
unfolding *eventually-principal* **by** *blast*
qed *fact*
qed

definition *nhdsin-sets* :: $'a$ *topology* $\Rightarrow 'a \Rightarrow 'a$ *set filter* **where**
nhdsin-sets X x $\equiv$ *derived-filter* {*U*. *openin X U* $\wedge$ $x \in U$} ($\supseteq$)

lemma *eventually-nhdsin-sets*:
assumes $x \in$ *topspace X*
shows *eventually P* (*nhdsin-sets X x*) $\longleftrightarrow$ ($\exists U.$ *openin X U* $\wedge$ $x \in U$ $\wedge$ ($\forall V.$ *openin X V* $\longrightarrow x \in V \longrightarrow V \subseteq U \longrightarrow$ *P V*))
proof −
have *h*:{*U*. *openin X U* $\wedge$ $x \in U$} $\neq$ {}
$\bigwedge a.\ a \in$ {*U*. *openin X U* $\wedge$ $x \in U$} $\Longrightarrow$ ($\supseteq$) *a a*

$\bigwedge a\ b\ c.\ a \in \{U.\ openin\ X\ U \wedge x \in U\} \Longrightarrow b \in \{U.\ openin\ X\ U \wedge x \in U\} \Longrightarrow c \in \{U.\ openin\ X\ U \wedge x \in U\} \Longrightarrow (\supseteq)\ a\ b \Longrightarrow (\supseteq)\ b\ c \Longrightarrow (\supseteq)\ a\ c$

$\bigwedge a\ b.\ a \in \{U.\ openin\ X\ U \wedge x \in U\} \Longrightarrow b \in \{U.\ openin\ X\ U \wedge x \in U\} \Longrightarrow \exists\, c\in\{U.\ openin\ X\ U \wedge x \in U\}.\ (\supseteq)\ a\ c \wedge (\supseteq)\ b\ c$

proof *safe*
fix *U V*
assume $x \in U$ $x \in V$ *openin X U openin X V*
then show $\exists\, W\in\{U.\ openin\ X\ U \wedge x \in U\}.\ W \subseteq U \wedge W \subseteq V$
using *openin-Int*[*of X U V*] **by** *auto*
qed(*use assms* **in** *fastforce*)+
show *?thesis*
unfolding *nhdsin-sets-def*
apply(*subst eventually-derived-filter*[*of* $\{U.\ openin\ X\ U \wedge x \in U\}$ $(\supseteq)$])
using *h* **apply** *blast*
apply *simp*
using *h*
apply *blast*
using *h*
apply *blast*
apply *fastforce*
done
qed

lemma *eventually-nhdsin-setsI*:
assumes $x \in topspace\ X$ $\bigwedge U.\ x \in U \Longrightarrow openin\ X\ U \Longrightarrow P\ U$
shows *eventually P* (*nhdsin-sets X x*)
using *assms* **by**(*auto simp*: *eventually-nhdsin-sets*)

lemma *nhdsin-sets-bot*[*simp*, *intro*]:
assumes $x \in topspace\ X$
shows $nhdsin\text{-}sets\ X\ x \neq \bot$
by(*auto simp*: *trivial-limit-def eventually-nhdsin-sets*[*OF assms*])

corollary *limitin-nhdsin-sets*: $limitin\ X\ xn\ x\ (nhdsin\text{-}sets\ X\ x) \longleftrightarrow x \in topspace\ X \wedge (\forall\, U.\ openin\ X\ U \longrightarrow x \in U \longrightarrow (\exists\, V.\ openin\ X\ V \wedge x \in V \wedge (\forall\, W.\ openin\ X\ W \longrightarrow x \in W \longrightarrow W \subseteq V \longrightarrow xn\ W \in U)))$
using *eventually-nhdsin-sets* **by**(*fastforce dest*: *limitin-topspace simp*: *limitin-def*)

lemma *closedin-limitin*:
assumes $T \subseteq topspace\ X$ $\bigwedge xn\ x.\ x \in topspace\ X \Longrightarrow (\bigwedge U.\ x \in U \Longrightarrow openin\ X\ U \Longrightarrow xn\ U \neq x) \Longrightarrow (\bigwedge U.\ x \in U \Longrightarrow openin\ X\ U \Longrightarrow xn\ U \in T) \Longrightarrow (\bigwedge U.\ xn\ U \in topspace\ X) \Longrightarrow limitin\ X\ xn\ x\ (nhdsin\text{-}sets\ X\ x) \Longrightarrow x \in T$
shows *closedin X T*
proof −
have *1*: $X\ derived\text{-}set\text{-}of\ T \subseteq T$
proof
fix *x*
assume $x{:}x \in X\ derived\text{-}set\text{-}of\ T$
hence $x'{:}x \in topspace\ X$

by (*simp add*: *in-derived-set-of*)
define *xn* **where** *xn* ≡ (λ*U*. *if x* ∈ *U* ∧ *openin X U then* (*SOME y*. *y* ≠ *x* ∧ *y* ∈ *T* ∧ *y* ∈ *U*) *else x*)
have *xn*: *xn U* ≠ *x xn U* ∈ *T xn U* ∈ *U* **if** *U*: *openin X U x* ∈ *U* **for** *U*
proof −
have (*SOME y*. *y* ≠ *x* ∧ *y* ∈ *T* ∧ *y* ∈ *U*) ≠ *x* ∧ (*SOME y*. *y* ≠ *x* ∧ *y* ∈ *T* ∧ *y* ∈ *U*) ∈ *T* ∧ (*SOME y*. *y* ≠ *x* ∧ *y* ∈ *T* ∧ *y* ∈ *U*) ∈ *U*
by(*rule someI-ex*,*insert x U*) (*auto simp*: *derived-set-of-def*)
thus *xn U* ≠ *x xn U* ∈ *T xn U* ∈ *U*
by(*auto simp*: *xn-def U*)
qed
hence *1*:⋀*U*. *x* ∈ *U* ⟹ *openin X U* ⟹ *xn U* ≠ *x* ⋀*U*. *x* ∈ *U* ⟹ *openin X U* ⟹ *xn U* ∈ *T*
by *simp-all*
moreover have *xn U* ∈ *topspace X* **for** *U*
proof(*cases x* ∈ *U* ∧ *openin X U*)
case *True*
with *assms 1* **show** *?thesis*
by *fast*
next
case *False*
with *x 1 derived-set-of-subset-topspace*[*of X T*] **show** *?thesis*
by(*auto simp*: *xn-def*)
qed
moreover have *limitin X xn x* (*nhdsin-sets X x*)
unfolding *limitin-nhdsin-sets*
proof *safe*
fix *U*
assume *U*: *openin X U x* ∈ *U*
then show ∃ *V*. *openin X V* ∧ *x* ∈ *V* ∧ (∀ *W*. *openin X W* ⟶ *x* ∈ *W* ⟶ *W* ⊆ *V* ⟶ *xn W* ∈ *U*)
using *xn* **by**(*fastforce intro*!: *exI*[**where** *x*=*U*])
qed(*use x derived-set-of-subset-topspace* **in** *fastforce*)
ultimately show *x* ∈ *T*
by(*rule assms*(*2*)[*OF x'*])
qed
thus *?thesis*
using *assms*(*1*) **by**(*auto intro*!: *closure-of-eq*[*THEN iffD1*] *simp*: *closure-of*)
qed

corollary *closedin-iff-limitin-eq*:
fixes *X* :: *'a topology*
shows *closedin X C*
⟷ *C* ⊆ *topspace X* ∧
(∀ *xi x* (*F* :: *'a set filter*). (∀ *i*. *xi i* ∈ *topspace X*) ⟶ *x* ∈ *topspace X*
⟶ (∀$_F$ *i in F*. *xi i* ∈ *C*) ⟶ *F* ≠ ⊥ ⟶ *limitin X xi x F* ⟶ *x* ∈ *C*)
proof
assume *C* ⊆ *topspace X* ∧
(∀ *xi x* (*F* :: *'a set filter*). (∀ *i*. *xi i* ∈ *topspace X*) ⟶ *x* ∈ *topspace X*

$\longrightarrow (\forall_F\ i\ in\ F.\ xi\ i \in C) \ \longrightarrow F \neq \bot \longrightarrow limitin\ X\ xi\ x\ F \longrightarrow x \in C)$
then show *closedin X C*
apply(*intro closedin-limitin*)
apply *blast*
by (*metis* (*mono-tags*, *lifting*) *nhdsin-sets-bot eventually-nhdsin-setsI*)
qed(*auto dest*: *limitin-closedin closedin-subset*)

lemma *closedin-iff-limitin-sequentially*:
assumes *first-countable X*
shows *closedin X S* $\longleftrightarrow S \subseteq topspace\ X \wedge (\forall \sigma\ l.\ range\ \sigma \subseteq S \wedge limitin\ X\ \sigma\ l\ sequentially \longrightarrow l \in S)$ (**is** *?lhs=?rhs*)
proof *safe*
assume $h{:}S \subseteq topspace\ X\ \forall \sigma\ l.\ range\ \sigma \subseteq S \wedge limitin\ X\ \sigma\ l\ sequentially \longrightarrow l \in S$
show *closedin X S*
proof(*rule closedin-limitin*)
fix *xu x*
assume *xu*: $\bigwedge U.\ x \in U \Longrightarrow openin\ X\ U \Longrightarrow xu\ U \in S\ \bigwedge U.\ xu\ U \in topspace\ X\ limitin\ X\ xu\ x\ (nhdsin\text{-}sets\ X\ x)$
then have $x{:}x \in topspace\ X$
by(*auto simp*: *limitin-topspace*)
then obtain *B* **where** *B*: $countable\ B\ \bigwedge V.\ V \in B \Longrightarrow openin\ X\ V$
$\bigwedge U.\ openin\ X\ U \Longrightarrow x \in U \Longrightarrow (\exists V{\in}B.\ x \in V \wedge V \subseteq U)$
using *assms first-countable-def* **by** *metis*
define B' **where** $B' \equiv B \cap \{U.\ x \in U\}$
have $B'\text{-}ne{:}B' \neq \{\}$
using $B'\text{-}def\ B(3)\ x$ **by** *fastforce*
have $cB'{:}countable\ B'$
using *B* **by**(*simp add*: $B'\text{-}def$)
have B': $\bigwedge V.\ V \in B' \Longrightarrow openin\ X\ V\ \bigwedge U.\ openin\ X\ U \Longrightarrow x \in U \Longrightarrow (\exists V{\in}B'.\ x \in V \wedge V \subseteq U)$
using $B\ B'\text{-}def$ **by** *fastforce+*
define *xn* **where** $xn \equiv (\lambda n.\ xu\ (\bigcap i{\le}n.\ (from\text{-}nat\text{-}into\ B'\ i)))$
have *xn-in-S*: $range\ xn \subseteq S$ **and** *limitin-xn*: *limitin X xn x sequentially*
proof −
have $1{:}\bigwedge n.\ openin\ X\ (\bigcap i{\le}n.\ (from\text{-}nat\text{-}into\ B'\ i))$
by (*auto simp*: $B'(1)\ B'\text{-}ne\ from\text{-}nat\text{-}into$)
have *2*: $\bigwedge n.\ x \in (\bigcap i{\le}n.\ (from\text{-}nat\text{-}into\ B'\ i))$
by (*metis* $B'\text{-}def\ B'\text{-}ne$ *INT-I Int-iff from-nat-into mem-Collect-eq*)
thus $range\ xn \subseteq S$
using *1* **by**(*auto simp*: *xn-def intro*!: *xu*)
show *limitin X xn x sequentially*
unfolding *limitin-sequentially*
proof *safe*
fix *U*
assume *U*: $openin\ X\ U\ x \in U$
then obtain *V* **where** *V*: $x \in V\ openin\ X\ V\ \bigwedge W.\ openin\ X\ W \Longrightarrow x \in W \Longrightarrow W \subseteq V \Longrightarrow xu\ W \in U$

by (*metis limitin-nhdsin-sets xu*(*3*))
then obtain *V′* **where** *V′*: *V′* ∈ *B′* *x* ∈ *V′* *V′* ⊆ *V*
using *B′* **by** *meson*
then obtain *N* **where** *N*: (⋂ *i*≤*N*. (*from-nat-into B′ i*)) ⊆ *V′*
by (*metis Inf-lower atMost-iff cB′ from-nat-into-surj image-iff order.refl*)
show ∃ *N*. ∀ *n*≥*N*. *xn n* ∈ *U*
proof(*safe intro*!: *exI*[**where** *x*=*N*])
fix *n*
assume [*arith*]:*n* ≥ *N*
show *xn n* ∈ *U*
unfolding *xn-def*
proof(*rule V*(*3*))
have (⋂ *i*≤*n*. (*from-nat-into B′ i*)) ⊆ (⋂ *i*≤*N*. (*from-nat-into B′ i*))
by *force*
also have ... ⊆ *V*
using *N V′* **by** *simp*
finally show ⋂ (*from-nat-into B′* ‘ {..*n*}) ⊆ *V* **.**
qed(*use 1 2* **in** *auto*)
qed
qed *fact*
qed
thus *x* ∈ *S*
using *h*(*2*) **by** *blast*
qed *fact*
qed(*auto simp*: *limitin-closedin range-subsetD dest*: *closedin-subset*)

1.6 A Characterization of Topology by Limit

lemma *topology-eq-filter*:
fixes *X* :: *′a topology*
assumes *topspace X* = *topspace Y*
and ⋀(*F* :: *′a set filter*) *xi x*. (⋀*i*. *xi i* ∈ *topspace X*) ⟹ *x* ∈ *topspace X* ⟹ *limitin X xi x F* ⟷ *limitin Y xi x F*
shows *X* = *Y*
unfolding *topology-eq-closedin closedin-iff-limitin-eq* **using** *assms* **by** *simp*

lemma *topology-eq-limit-sequentially*:
assumes *topspace X* = *topspace Y*
and *first-countable X first-countable Y*
and ⋀*xn x*. (⋀*n*. *xn i* ∈ *topspace X*) ⟹ *x* ∈ *topspace X* ⟹ *limitin X xn x sequentially* ⟷ *limitin Y xn x sequentially*
shows *X* = *Y*
unfolding *topology-eq-closedin closedin-iff-limitin-sequentially*[*OF assms*(*2*)] *closedin-iff-limitin-sequentially*[
assms(*3*)]
by (*metis dual-order.trans limitin-topspace range-subsetD assms*(*1*,*4*))

1.7 A Characterization of Open Sets by Limit

corollary *openin-limitin*:
assumes *U* ⊆ *topspace X* ⋀*xi x*. *x* ∈ *topspace X* ⟹ (⋀*i*. *xi i* ∈ *topspace X*)

⟹ *limitin X xi x* (*nhdsin-sets X x*) ⟹ *x* ∈ *U* ⟹ ∀$_F$ *i in* (*nhdsin-sets X x*). *xi i* ∈ *U*
shows *openin X U*
unfolding *openin-closedin-eq*
proof(*safe intro*!: *assms*(*1*) *closedin-limitin*)
fix *xi x*
assume *h*:*x* ∈ *topspace X* ∀ *V*. *x* ∈ *V* ⟶ *openin X V* ⟶ *xi V* ∈ *topspace X* − *U*
∀ *V*. *xi V* ∈ *topspace X limitin X xi x* (*nhdsin-sets X x*) *x* ∈ *U*
show *False*
using *assms*(*2*)[*OF h*(*1*,*3*,*4*,*5*)[*rule-format*]] *h*(*2*)
by(*auto simp*: *eventually-nhdsin-sets*[*OF h*(*1*)])
qed

corollary *openin-iff-limitin-eq*:
fixes *X* :: *'a topology*
shows *openin X U* ⟷ *U* ⊆ *topspace X* ∧ (∀ *xi x* (*F* :: *'a set filter*). (∀ *i*. *xi i* ∈ *topspace X*) ⟶ *x* ∈ *U* ⟶ *limitin X xi x F* ⟶ (∀$_F$ *i in F*. *xi i* ∈ *U*))
by(*auto intro*!: *openin-limitin openin-subset simp*: *limitin-def*)

corollary *limitin-openin-sequentially*:
assumes *first-countable X*
shows *openin X U* ⟷ *U* ⊆ *topspace X* ∧ (∀ *xn x*. *x* ∈ *U* ⟶ *limitin X xn x sequentially* ⟶ (∃ *N*. ∀ *n*≥*N*. *xn n* ∈ *U*))
unfolding *openin-closedin-eq closedin-iff-limitin-sequentially*[*OF assms*]
apply *safe*
apply (*simp add*: *assms closedin-iff-limitin-sequentially limitin-sequentially openin-closedin*)
apply (*simp add*: *limitin-sequentially*)
apply *blast*
done

lemma *upper-semicontinuous-map-limsup-iff*:
fixes *f* :: *'a* ⇒ (*'b* :: {*complete-linorder*,*linorder-topology*})
assumes *first-countable X*
shows *upper-semicontinuous-map X f* ⟷ (∀ *xn x*. *limitin X xn x sequentially* ⟶ *limsup* (λ*n*. *f* (*xn n*)) ≤ *f x*)
unfolding *upper-semicontinuous-map-def*
proof *safe*
fix *xn x*
assume *h*:∀ *a*. *openin X* {*x* ∈ *topspace X*. *f x* < *a*} *limitin X xn x sequentially*
show *limsup* (λ*n*. *f* (*xn n*)) ≤ *f x*
unfolding *Limsup-le-iff eventually-sequentially*
proof *safe*
fix *y*
assume *y*:*f x* < *y*
show ∃ *N*. ∀ *n*≥*N*. *f* (*xn n*) < *y*
proof(*rule ccontr*)
assume ∄ *N*. ∀ *n*≥*N*. *f* (*xn n*) < *y*

```
      then have hc:⋀N. ∃n≥N. f (xn n) ≥ y
        using linorder-not-less by blast
      define a :: nat ⇒ nat where a ≡ rec-nat (SOME n. f (xn n) ≥ y) (λn an.
SOME m. m > an ∧ f (xn m) ≥ y)
      have strict-mono a
      proof(rule strict-monoI-Suc)
        fix n
        have [simp]:a (Suc n) = (SOME m. m > a n ∧ f (xn m) ≥ y)
          by(auto simp: a-def)
        show a n < a (Suc n)
          by simp (metis (mono-tags, lifting) Suc-le-lessD hc someI)
      qed
      have ∗:f (xn (a n)) ≥ y for n
      proof(cases n)
        case 0
        then show ?thesis
          using hc[of 0] by(auto simp: a-def intro!: someI-ex)
      next
        case (Suc n')
        then show ?thesis
          by(simp add: a-def) (metis (mono-tags, lifting) Suc-le-lessD hc someI-ex)
      qed
      have ∃N. ∀n≥N. (xn ∘ a) n ∈ {x ∈ topspace X. f x < y}
       using limitin-subsequence[OF ‹strict-mono a› h(2)] h(1)[rule-format,of y] y
        by(fastforce simp: limitin-sequentially)
      with ∗ show False
        using leD by auto
    qed
  qed
next
  fix a
  assume h: ∀xn x. limitin X xn x sequentially ⟶ limsup (λn. f (xn n)) ≤ f x
  show openin X {x ∈ topspace X. f x < a}
    unfolding limitin-openin-sequentially[OF assms]
  proof safe
    fix x xn
    assume h':limitin X xn x sequentially x ∈ topspace X f x < a
    with h have limsup (λn. f (xn n)) ≤ f x
      by auto
    with h'(3) obtain N where N:⋀n. n≥N ⟹ f (xn n) < a
      by(auto simp: Limsup-le-iff eventually-sequentially)
    obtain N' where N': ⋀n. n≥N' ⟹ xn n ∈ topspace X
      by (meson h'(1) limitin-sequentially openin-topspace)
    thus ∃N. ∀n≥N. xn n ∈ {x ∈ topspace X. f x < a}
      using h'(3) N by(auto intro!: exI[where x=max N N'])
  qed
qed
```

1.8 Lemmas for Upper/Lower-Semi Continuous Maps

lemma *upper-semicontinuous-map-limsup-real*:
  fixes $f :: 'a \Rightarrow real$
  assumes *first-countable X*
  shows *upper-semicontinuous-map X f* $\longleftrightarrow$ $(\forall\, xn\ x.\ limitin\ X\ xn\ x\ sequentially \longrightarrow limsup\ (\lambda n.\ f\ (xn\ n)) \le f\ x)$
  unfolding *upper-semicontinuous-map-real-iff upper-semicontinuous-map-limsup-iff* [*OF assms*] **by** *simp*

lemma *lower-semicontinuous-map-liminf-iff*:
  fixes $f :: 'a \Rightarrow ('b :: \{complete\text{-}linorder, linorder\text{-}topology\})$
  assumes *first-countable X*
  shows *lower-semicontinuous-map X f* $\longleftrightarrow$ $(\forall\, xn\ x.\ limitin\ X\ xn\ x\ sequentially \longrightarrow f\ x \le liminf\ (\lambda n.\ f\ (xn\ n)))$
  unfolding *lower-semicontinuous-map-def*
proof *safe*
  fix *xn x*
  assume $h{:}\forall\, a.\ openin\ X\ \{x \in topspace\ X.\ a < f\ x\}\ limitin\ X\ xn\ x\ sequentially$
  show $f\ x \le liminf\ (\lambda n.\ f\ (xn\ n))$
    unfolding *le-Liminf-iff eventually-sequentially*
  proof *safe*
    fix *y*
    assume $y{:}y < f\ x$
    show $\exists\, N.\ \forall\, n{\ge}N.\ y < f\ (xn\ n)$
    proof(*rule ccontr*)
      assume $\nexists\, N.\ \forall\, n{\ge}N.\ y < f\ (xn\ n)$
      then have $hc{:}\bigwedge N.\ \exists\, n{\ge}N.\ y \ge f\ (xn\ n)$
        by (*meson verit-comp-simplify1* (*3*))
      define $a :: nat \Rightarrow nat$ **where** $a \equiv rec\text{-}nat\ (SOME\ n.\ f\ (xn\ n) \le y)\ (\lambda n\ an.\ SOME\ m.\ m > an \wedge f\ (xn\ m) \le y)$
      have *strict-mono a*
      proof(*rule strict-monoI-Suc*)
        fix *n*
        have [*simp*]: $a\ (Suc\ n) = (SOME\ m.\ m > a\ n \wedge f\ (xn\ m) \le y)$
          by(*auto simp*: *a-def*)
        show $a\ n < a\ (Suc\ n)$
          by *simp* (*metis* (*no-types*, *lifting*) *Suc-le-lessD* ‹$\nexists\, N.\ \forall\, n{\ge}N.\ y < f\ (xn\ n)$› *not-le someI-ex*)
      qed
      have $*{:}f\ (xn\ (a\ n)) \le y$ **for** *n*
      proof(*cases n*)
        case *0*
        then show *?thesis*
          using *hc*[*of 0*] **by**(*auto simp*: *a-def intro*!: *someI-ex*)
      next
        case (*Suc n'*)
        then show *?thesis*
          by(*simp add*: *a-def*) (*metis* (*mono-tags*, *lifting*) *Suc-le-lessD hc someI-ex*)
      qed

```
      have ∃ N. ∀ n≥N. (xn ∘ a) n ∈ {x ∈ topspace X. f x > y}
       using limitin-subsequence[OF ‹strict-mono a› h(2)] h(1)[rule-format,of y] y
        by(fastforce simp: limitin-sequentially)
      with ∗ show False
        using leD by auto
    qed
  qed
next
  fix a
  assume h: ∀ xn x. limitin X xn x sequentially ⟶ f x ≤ liminf (λn. f (xn n))
  show openin X {x ∈ topspace X. a < f x}
    unfolding limitin-openin-sequentially[OF assms]
  proof safe
    fix x xn
    assume h′:limitin X xn x sequentially x ∈ topspace X f x > a
    with h have f x ≤ liminf (λn. f (xn n))
      by auto
    with h′(3) obtain N where N:⋀n. n≥N ⟹ f (xn n) > a
      by(auto simp: le-Liminf-iff eventually-sequentially)
    obtain N′ where N′: ⋀n. n≥N′ ⟹ xn n ∈ topspace X
      by (meson h′(1) limitin-sequentially openin-topspace)
    thus ∃ N. ∀ n≥N. xn n ∈ {x ∈ topspace X. f x > a}
      using h′(3) N by(auto intro!: exI[where x=max N N′])
  qed
qed

lemma lower-semicontinuous-map-liminf-real:
  fixes f :: ′a ⇒ real
  assumes first-countable X
   shows lower-semicontinuous-map X f ⟷ (∀ xn x. limitin X xn x sequentially
⟶ f x ≤ liminf (λn. f (xn n)))
 unfolding lower-semicontinuous-map-real-iff lower-semicontinuous-map-liminf-iff[OF
assms] by simp

end
```

2 Alaoglu's Theorem

```
theory Alaoglu-Theorem
  imports Lemmas-Levy-Prokhorov
          Riesz-Representation.Riesz-Representation
begin
```

We prove (a special case of) Alaoglu's theorem for the space of continuous functions. We refer to Section 9 of the lecture note by Heil [1].

2.1 Metrizability of the Space of Uniformly Bounded Positive Linear Functionals

lemma *metrizable-functional*:
  fixes *X* :: *'a topology* **and** *r* :: *real*
  defines *prod-space* ≡ *powertop-real* (*mspace* (*cfunspace X euclidean-metric*))
  defines *B* ≡ {*φ*∈*topspace prod-space*. *φ* (*λx*∈*topspace X*. *1*) ≤ *r* ∧ *positive-linear-functional-on-CX X φ*}
  defines Φ ≡ *subtopology prod-space B*
  assumes *compact*: *compact-space X* **and** *met*: *metrizable-space X*
  shows *metrizable-space* Φ
proof(*cases X* = *trivial-topology*)
  case *True*
  hence *metrizable-space prod-space*
   by(*auto simp*: *prod-space-def metrizable-space-product-topology metrizable-space-euclidean intro*!: *countable-finite*)
  thus *?thesis*
    using Φ*-def metrizable-space-subtopology* **by** *blast*
next
  case *X-ne*:*False*
  have *Haus*: *Hausdorff-space X*
    using *met metrizable-imp-Hausdorff-space* **by** *blast*
  consider *r* ≥ *0* | *r* < *0*
    by *fastforce*
  then show *?thesis*
  proof *cases*
    case *r*:*1*
    have *B*: *B* ⊆ *topspace prod-space*
      by(*auto simp*: *B-def*)
     have *ext-eq*: ⋀*f*::'*a* ⇒ *real*. *f* ∈ *mspace* (*cfunspace X euclidean-metric*) ⟹ (*λx*∈*topspace X*. *f x*) = *f*
      by (*auto simp*: *extensional-def*)
    have *met1*: *metrizable-space* (*mtopology-of* (*cfunspace X euclidean-metric*))
     by (*metis Metric-space.metrizable-space-mtopology Metric-space-mspace-mdist mtopology-of-def*)
    have *separable-space* (*mtopology-of* (*cfunspace X* (*euclidean-metric* :: *real metric*)))
    proof −
     have *separable-space* (*mtopology-of* (*cfunspace X* (*Met-TC.Self* :: *real metric*)))
      using *Met-TC.Metric-space-axioms Met-TC.separable-space-iff-second-countable*
       by(*auto intro*!: *Metric-space.separable-space-cfunspace*[*OF - - - met compact*])
      thus *?thesis*
        by (*simp add*: *Met-TC.Self-def euclidean-metric-def*)
    qed
    then obtain *F* **where** *dense*:*mdense-of* (*cfunspace X* (*euclidean-metric* :: *real metric*)) *F* **and** *F-count*: *countable F*
      using *separable-space-def2* **by** *blast*
     have *infinite* (*topspace* (*mtopology-of* (*cfunspace X* (*euclidean-metric* :: *real metric*))))
    proof(*rule infinite-super*[**where** *S*=(*λn*::*nat*. *λx*∈*topspace X*. *real n*) ‘ *UNIV*])

show *infinite* (*range* (λ*n*. λ*x*∈*topspace X*. *real n*))
proof(*intro range-inj-infinite inj-onI*)
fix *n m*
assume *h*:(λ*x*∈*topspace X*. *real n*) = (λ*x*∈*topspace X*. *real m*)
from *X-ne* **obtain** *x* **where** *x* ∈ *topspace X* **by** *fastforce*
with *fun-cong*[*OF h*,*of x*] **show** *n* = *m*
by *simp*
qed
qed(*auto simp*: *bounded-iff*)
from *countable-as-injective-image*[*OF F-count dense-in-infinite*[*OF metrizable-imp-t1-space*[*OF met1*] *this dense*]]
obtain *gn* :: *nat* ⇒ - **where** *gn*: *F* = *range gn inj gn*
by *blast*
then have *gn-in*: ⋀*n*. *gn n* ∈ *F* ⋀*n*. *gn n* ∈ *mspace* (*cfunspace X euclidean-metric*)
using *dense-in-subset*[*OF dense*] **by** *auto*
hence *gn-ext*: ⋀*n*. (λ*x*∈*topspace X*. *gn n x*) = *gn n*
by(*auto intro*!: *ext-eq*)
define *d* :: [('*a* ⇒ *real*) ⇒ *real*, ('*a* ⇒ *real*) ⇒ *real*] ⇒ *real*
where *d* ≡ (λφ ψ. (∑ *n*. (*1* / *2*) ^ *n* ∗ *mdist* (*capped-metric 1 euclidean-metric*)
(φ (λ*x*∈*topspace X*. *gn n x*)) (ψ (λ*x*∈*topspace X*. *gn n x*))))
have *smble*[*simp*]: *summable* (λ*n*. (*1* / *2*) ^ *n* ∗ *mdist* (*capped-metric 1* (*euclidean-metric* :: *real metric*)) (*a n*) (*b n*))
for *a b*
by(*rule summable-comparison-test'*[**where** *N*=*0* **and** *g*=λ*n*. (*1* / *2*) ^ *n* ∗ *1*]) (*auto intro*!: *mdist-capped*)
interpret *d*: *Metric-space topspace* Φ *d*
proof
show ⋀*x y*. *0* ≤ *d x y*
by(*auto intro*!: *suminf-nonneg simp*: *d-def*)
show ⋀*x y*. *d x y* = *d y x*
by(*auto simp*: *d-def simp*: *mdist-commute*)
next
fix φ ψ
assume *h*:φ ∈ *topspace* Φ ψ ∈ *topspace* Φ
show *d* φ ψ = *0* ⟷ φ = ψ
proof
assume *d* φ ψ = *0*
then have ∀ *n*. (*1* / *2*) ^ *n* ∗ *mdist* (*capped-metric 1 euclidean-metric*)
(φ (λ*x*∈*topspace X*. *gn n x*)) (ψ (λ*x*∈*topspace X*. *gn n x*)) = *0*
by(*intro suminf-eq-zero-iff*[*THEN iffD1*]) (*auto simp*: *d-def*)
hence *eq*:⋀*n*. φ (λ*x*∈*topspace X*. *gn n x*) = ψ (λ*x*∈*topspace X*. *gn n x*)
by *simp*
show φ = ψ
proof
fix *f*
consider *f* ∉ *mspace* (*cfunspace X* (*euclidean-metric* :: *real metric*))

| $f \in$ *mspace* (*cfunspace* X (*euclidean-metric* :: *real metric*))
by *blast*
thus $\varphi\ f = \psi\ f$
proof *cases*
case *1*
then show *?thesis*
using *h* **by**(*auto simp*: Φ*-def prod-space-def PiE-def extensional-def simp del*: *mspace-cfunspace*)
next
case *f*:*2*
then have *positive-linear-functional-on-CX* X φ *positive-linear-functional-on-CX* X ψ
using *h* **by**(*auto simp*: Φ*-def topspace-subtopology-subset*[*OF B*] *B-def*)
from *Riesz-representation-real-compact-metrizable*[*OF compact met this*(*1*)]
Riesz-representation-real-compact-metrizable[*OF compact met this*(*2*)]
obtain *N L* **where**
N: *sets N* = *sets* (*borel-of X*) *finite-measure N*
$\bigwedge f$. *continuous-map X euclideanreal f* $\Longrightarrow$ φ (*restrict f* (*topspace X*)) = $integral^L$ *N f*
and *L*: *sets L* = *sets* (*borel-of X*) *finite-measure L*
$\bigwedge f$. *continuous-map X euclideanreal f* $\Longrightarrow$ ψ (*restrict f* (*topspace X*)) = $integral^L$ *L f*
by *auto*
have *f-ext*:($\lambda x \in$*topspace X*. *f x*) = *f*
using *f* **by** (*auto simp*: *extensional-def*)
have $\varphi\ f = \varphi$ ($\lambda x \in$*topspace X*. *f x*)
by(*simp add*: *f-ext*)
also have ... = $integral^L$ *N f*
using *f* **by**(*auto intro*!: *N*)
also have ... = $integral^L$ *L f*
proof(*rule finite-measure-integral-eq-dense*[**where** *F*=*F* **and** *X*=*X*])
fix *g*
assume $g \in F$
then obtain *n* **where** *n*:*g* = *gn n*
using *gn* **by** *fast*
hence $integral^L$ *N g* = $integral^L$ *N* (*gn n*)
by *simp*
also have ... = φ ($\lambda x \in$*topspace X*. *gn n x*)
using *gn-in* **by**(*auto intro*!: *N*(*3*)[*symmetric*])
also have ... = $integral^L$ *L g*
using *gn-in* **by**(*auto simp*: *eq n intro*!: *L*(*3*))
finally show $integral^L$ *N g* = $integral^L$ *L g* **.**
qed(*use N L dense f* **in** *auto*)
also have ... = ψ ($\lambda x \in$*topspace X*. *f x*)
using *f* **by**(*auto intro*!: *L*(*3*)[*symmetric*])
also have ... = $\psi\ f$
by(*simp add*: *f-ext*)
finally show *?thesis* **.**

```
        qed
      qed
    qed (auto simp add: d-def capped-metric-mdist)
  next
    fix φ1 φ2 φ3
    assume h: φ1 ∈ topspace Φ φ2 ∈ topspace Φ φ3 ∈ topspace Φ
    show d φ1 φ3 ≤ d φ1 φ2 + d φ2 φ3
    proof −
    have d φ1 φ3 ≤ (∑ n. (1 / 2) ^ n ∗ mdist (capped-metric 1 euclidean-metric)
                                        (φ1 (λx∈topspace X. gn n x)) (φ2
(λx∈topspace X. gn n x))
                        + (1 / 2) ^ n ∗ mdist (capped-metric 1 euclidean-metric)
                                        (φ2 (λx∈topspace X. gn n x)) (φ3
(λx∈topspace X. gn n x)))
          by(auto intro!: suminf-le mdist-triangle summable-add[OF smble sm-
ble,simplified distrib-left[symmetric]]
          simp: d-def distrib-left[symmetric])
      also have ... = d φ1 φ2 + d φ2 φ3
        by(simp add: suminf-add d-def)
      finally show ?thesis .
    qed
  qed
  have Φ = d.mtopology
    unfolding topology-eq
  proof safe
    have continuous-map d.mtopology (subtopology prod-space B) id
      unfolding continuous-map-in-subtopology prod-space-def id-apply image-id
continuous-map-componentwise
    proof safe
      fix f :: 'a ⇒ real
      assume f: f ∈ mspace (cfunspace X (euclidean-metric))
      hence f-ext: (λx∈topspace X. f x) = f
        by(auto intro!: ext-eq)
      show continuous-map d.mtopology euclideanreal (λx. x f)
        unfolding continuous-map-iff-limit-seq[OF d.first-countable-mtopology]
      proof safe
        fix φn φ
        assume φ-limit:limitin d.mtopology φn φ sequentially
        have (λn. φn n f) ⟶ φ f
        proof(rule LIMSEQ-I)
          fix e :: real
          assume e: e > 0
          from f mdense-of-def3[THEN iffD1,OF dense] obtain fn where fn:
            ⋀n. fn n ∈ F limitin (mtopology-of (cfunspace X euclidean-metric)) fn
f sequentially
            by fast
          with f dense-in-subset[OF dense] have fn-ext:⋀n. (λx∈topspace X. fn n
x) = fn n
            by(intro ext-eq) auto
```

define *a0* **where** $a0 \equiv (SOME\ n.\ \forall x{\in}topspace\ X.\ |fn\ n\ x - f\ x| \le (1 / 3) * (1 / (r + 1)) * e)$
have $a0{:}\forall x{\in}topspace\ X.\ |fn\ a0\ x - f\ x| \le (1 / 3) * (1 / (r + 1)) * e$
unfolding *a0-def*
proof(*rule someI-ex*)
have $\bigwedge e.\ e > 0 \Longrightarrow \exists N.\ \forall n{\ge}N.\ mdist\ (cfunspace\ X\ euclidean\text{-}metric)\ (fn\ n)\ f < e$
by (*metis Metric-space.limit-metric-sequentially Metric-space-mspace-mdist fn*(*2*) *mtopology-of-def*)
from $this[of\ ((1 / 3) * (1 / (r + 1)) * e)]$
obtain *N* **where** $N{:}\ \bigwedge n.\ n \ge N \Longrightarrow mdist\ (cfunspace\ X\ euclidean\text{-}metric)\ (fn\ n)\ f < ((1 / 3) * (1 / (r + 1)) * e)$
using *e r* **by** *auto*
hence $mdist\ (cfunspace\ X\ euclidean\text{-}metric)\ (fn\ N)\ f \le ((1 / 3) * (1 / (r + 1)) * e)$
by *fastforce*
from *mdist-cfunspace-imp-mdist-le*[*OF - - this*]
have $le{:}\bigwedge x.\ x \in topspace\ X \Longrightarrow |fn\ N\ x - f\ x| \le ((1 / 3) * (1 / (r + 1)) * e)$
using *fn*(*1*)[*of N*] *dense-in-subset*[*OF dense*] *f dist-real-def* **by** *auto*
thus $\exists n.\ \forall x{\in}topspace\ X.\ |fn\ n\ x - f\ x| \le 1 / 3 * (1 / (r + 1)) * e$
by(*auto intro*!: *exI*[**where** *x*=*N*])
qed
obtain *l* **where** $l{:}\ fn\ a0 = gn\ l$
using *fn gn* **by** *blast*
have $\bigwedge e.\ e > 0 \Longrightarrow \exists N.\ \forall n{\ge}N.\ \varphi n\ n \in topspace\ \Phi \wedge d\ (\varphi n\ n)\ \varphi < e$
using *φ-limit* **by**(*fastforce simp*: *mtopology-of-def d.limit-metric-sequentially*)
from $this[of\ (1 / 2)\ \hat{}\ l * (1 / 3) * min\ 3\ e]\ e$
obtain *N* **where** $N{:}\ \bigwedge n.\ n \ge N \Longrightarrow \varphi n\ n \in topspace\ \Phi$
$\bigwedge n.\ n \ge N \Longrightarrow d\ (\varphi n\ n)\ \varphi < (1 / 2)\ \hat{}\ l * (1 / 3) * min\ 3\ e$
by *auto*
show $\exists no.\ \forall n{\ge}no.\ norm\ (\varphi n\ n\ f - \varphi\ f) < e$
proof(*safe intro*!: *exI*[**where** *x*=*N*])
fix *n*
assume $n{:}\ N \le n$
have $norm\ (\varphi n\ n\ f - \varphi\ f) \le |\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| + |\varphi\ (fn\ a0) - \varphi\ f| + |\varphi n\ n\ (fn\ a0) - \varphi n\ n\ f|$
by *fastforce*
also have $... < (1 / 3) * e + (1 / 3) * e + (1 / 3) * e$
proof −
have $1{:}\ |\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| < (1 / 3) * e$
proof(*rule ccontr*)
assume $\neg\ |\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| < 1 / 3 * e$
then have $1{:}|\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| \ge (1 / 3) * e$
by *linarith*
have $le1{:}\ |\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| < 1$
proof (*rule ccontr*)
assume $\neg\ |\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| < 1$
then have $contr{:}|\varphi n\ n\ (fn\ a0) - \varphi\ (fn\ a0)| \ge 1$

```
          by linarith
        consider e > 3 | e ≤ 3
          by fastforce
        then show False
        proof cases
          case 1
          with N[OF n] have d (φn n) φ < (1 / 2) ^ l
            by simp
          also have ... = (∑m. if m = l then (1 / 2) ^ l else 0)
            using suminf-split-initial-segment[where f=λm. if m = l then
(1 / 2) ^ l else (0 :: real) and k=Suc l]
            by simp
          also have ... ≤ d (φn n) φ
            unfolding d-def
          proof(rule suminf-le)
            fix m
            show (if m = l then (1 / 2) ^ l else 0)
                ≤ (1 / 2) ^ m ∗ mdist (capped-metric 1 euclidean-metric)
                                    (φn n (restrict (gn m) (topspace X)))
                                    (φ (restrict (gn m) (topspace X)))
                  using contr by(auto simp: l gn-ext capped-metric-mdist
dist-real-def)
          qed auto
          finally show False
            by blast
        next
          case 2
          then have (1 / 2) ^ l ∗ (1 / 3) ∗ min 3 e ≤ (1 / 2)^l
            by simp
          also have ... = (∑m. if m = l then (1 / 2) ^ l else 0)
            using suminf-split-initial-segment[where f=λm. if m = l then
(1 / 2) ^ l else (0 :: real) and k=Suc l]
            by simp
          also have ... ≤ d (φn n) φ
            unfolding d-def
          proof(rule suminf-le)
            fix m
            show (if m = l then (1 / 2) ^ l else 0)
                ≤ (1 / 2) ^ m ∗ mdist (capped-metric 1 euclidean-metric)
                                    (φn n (restrict (gn m) (topspace X)))
                                    (φ (restrict (gn m) (topspace X)))
                  using contr by(auto simp: l gn-ext capped-metric-mdist
dist-real-def)
          qed auto
          also have ... < (1 / 2) ^ l ∗ (1 / 3) ∗ min 3 e
            by(rule N[OF n])
          finally show False by simp
        qed
      qed
```

```
        hence mdist1: mdist (capped-metric 1 euclidean-metric)
                        (φn n (restrict (gn l) (topspace X)))
                        (φ (restrict (gn l) (topspace X)))
                   = |φn n (fn a0) − φ (fn a0)|
          by(auto simp: capped-metric-mdist dist-real-def gn-ext l)
        have (1 / 2) ^ l ∗ (1 / 3) ∗ min 3 e ≤ (1 / 2) ^ l ∗ (1 / 3) ∗ e
          using e by simp
      also have ... = (∑m. if m = l then (1 / 2) ^ l ∗ (1 / 3) ∗ e else 0)
          using suminf-split-initial-segment[where f=λm. if m = l then (1
/ 2) ^ l ∗ (1 / 3) ∗ e else 0 and k=Suc l]
          by simp
        also have ... ≤ d (φn n) φ
          using 1 le1 by (fastforce simp: mdist1 d-def intro!: suminf-le)
        finally show False
          using N[OF n] by linarith
      qed
      have 2: |φ (fn a0) − φ f| ≤ (1 / 3) ∗ e
      proof −
        from limitin-topspace[OF φ-limit,simplified]
        have plf:positive-linear-functional-on-CX X φ
          by(simp add: Φ-def B-def)
        from Riesz-representation-real-compact-metrizable[OF compact met
this]
        obtain N where N: sets N = sets (borel-of X) finite-measure N
          ⋀f. continuous-map X euclideanreal f ⟹ φ (restrict f (topspace
X)) = integral^L N f
          by blast
        hence space-N: space N = topspace X
          by(auto cong: sets-eq-imp-space-eq simp: space-borel-of)
        interpret N: finite-measure N by fact
        have [measurable]: fn a0 ∈ borel-measurable N f ∈ borel-measurable
N
            using continuous-map-measurable[of X euclideanreal] fn(1) f
dense-in-subset[OF dense]
          by(auto simp: measurable-cong-sets[OF N(1) refl]
             intro!: continuous-map-measurable[of X euclideanreal,simplified
borel-of-euclidean])
      have φ (fn a0) − φ f = φ (λx∈topspace X. fn a0 x) − φ (λx∈topspace
X. f x)
          by(simp add: fn-ext f-ext)
        also have ... = φ (λx∈topspace X. fn a0 x) + φ (λx∈topspace X.
− f x)
        using f by(auto intro!: pos-lin-functional-on-CX-compact-lin(1)[OF
plf compact,of - −1,simplified,symmetric])
        also have ... = φ (λx∈topspace X. fn a0 x + − f x)
          by(rule pos-lin-functional-on-CX-compact-lin(2)[symmetric])
            (use fn(1) f dense-in-subset[OF dense] plf compact in auto)
        also have ... = φ (λx∈topspace X. fn a0 x − f x)
          by simp
```

also have ... = ($\int x.$ *fn a0 x* − *f x* ∂*N*)
using *fn*(*1*) *f dense-in-subset*[*OF dense*] **by**(*auto intro*!: *N*(*3*) *continuous-map-diff*)
finally have |φ (*fn a0*) − φ *f*| = |($\int x.$ *fn a0 x* − *f x* ∂*N*)|
by *simp*
also have ... ≤ ($\int x.$ |*fn a0 x* − *f x*| ∂*N*)
by(*rule integral-abs-bound*)
also have ... ≤ ($\int x.$ (*1* / *3*) ∗ (*1* / (*r* + *1*)) ∗ *e* ∂*N*)
by(*rule Bochner-Integration.integral-mono*,*insert a0*)
(*auto intro*!: *N.integrable-const-bound*[**where** *B*=(*1* / *3*) ∗ (*1* / (*r* + *1*)) ∗ *e*] *simp*: *space-N*)
also have ... = (*1* / *3*) ∗ *e* ∗ ((*1* / (*r* + *1*) ∗ *measure N* (*space N*)))
by *simp*
also have ... ≤ (*1* / *3*) ∗ *e*
proof −
have *measure N* (*space N*) = ($\int x.$ *1* ∂*N*)
by *simp*
also have ... = φ (λ*x*∈*topspace X*. *1*)
by(*intro N*(*3*)[*symmetric*]) *simp*
also have ... ≤ *r*
using *limitin-topspace*[*OF* φ*-limit*,*simplified*] **by**(*auto simp*: Φ*-def B-def*)
finally have (*1* / (*r* + *1*) ∗ *measure N* (*space N*)) ≤ *1*
using *r* **by** *simp*
thus *?thesis*
unfolding *mult-le-cancel-left2* **using** *e* **by** *auto*
qed
finally show *?thesis* **.**
qed
have *3*: |φ*n n* (*fn a0*) − φ*n n f*| ≤ (*1* / *3*) ∗ *e*
proof −
have *plf*:*positive-linear-functional-on-CX X* (φ*n n*)
using *N*(*1*)[*OF n*] **by**(*simp add*: Φ*-def B-def*)
from *Riesz-representation-real-compact-metrizable*[*OF compact met this*]
obtain *N* **where** *N′*: *sets N* = *sets* (*borel-of X*) *finite-measure N*
⋀*f*. *continuous-map X euclideanreal f* ⟹ φ*n n* (*restrict f* (*topspace X*)) = $integral^L$ *N f*
by *blast*
hence *space-N*: *space N* = *topspace X*
by(*auto cong*: *sets-eq-imp-space-eq simp*: *space-borel-of*)
interpret *N*: *finite-measure N* **by** *fact*
have [*measurable*]: *fn a0* ∈ *borel-measurable N f* ∈ *borel-measurable N*
using *continuous-map-measurable*[*of X euclideanreal*] *fn*(*1*) *f dense-in-subset*[*OF dense*]
by(*auto simp*: *measurable-cong-sets*[*OF N′*(*1*) *refl*]
intro!: *continuous-map-measurable*[*of X euclideanreal*,*simplified borel-of-euclidean*])

have φn *n* (*fn a0*) $-$ φn *n f* $=$ φn *n* ($\lambda x \in$*topspace X*. *fn a0 x*) $-$ φn *n* ($\lambda x \in$*topspace X*. *f x*)
by(*simp add*: *fn-ext f-ext*)
also have ... $=$ φn *n* ($\lambda x \in$*topspace X*. *fn a0 x*) $+$ φn *n* ($\lambda x \in$*topspace X*. $-$ *f x*)
using *f* **by**(*auto intro*!: *pos-lin-functional-on-CX-compact-lin*(*1*)[*OF plf compact*,*of* - $-$*1*,*simplified*,*symmetric*])
also have ... $=$ φn *n* ($\lambda x \in$*topspace X*. *fn a0 x* $+$ $-$ *f x*)
by(*rule pos-lin-functional-on-CX-compact-lin*(*2*)[*symmetric*])
(*use fn*(*1*) *plf compact f dense-in-subset*[*OF dense*] **in** *auto*)
also have ... $=$ φn *n* ($\lambda x \in$*topspace X*. *fn a0 x* $-$ *f x*)
by *simp*
also have ... $=$ ($\int$ *x*. *fn a0 x* $-$ *f x* ∂N)
using *fn*(*1*) *f dense-in-subset*[*OF dense*] **by**(*auto intro*!: *N'*(*3*) *continuous-map-diff*)
finally have $|\varphi n$ *n* (*fn a0*) $-$ φn *n f*$|$ $=$ $|$($\int$ *x*. *fn a0 x* $-$ *f x* ∂N)$|$
by *simp*
also have ... $\le$ ($\int$ *x*. $|$*fn a0 x* $-$ *f x*$|$ ∂N)
by(*rule integral-abs-bound*)
also have ... $\le$ ($\int$ *x*. (*1* / *3*) $*$ (*1* / (*r* $+$ *1*)) $*$ *e* ∂N)
by(*rule Bochner-Integration.integral-mono*,*insert a0*)
(*auto intro*!: *N.integrable-const-bound*[**where** *B*=(*1* / *3*) $*$ (*1* / (*r* $+$ *1*)) $*$ *e*] *simp*: *space-N*)
also have ... $=$ (*1* / *3*) $*$ *e* $*$ ((*1* / (*r* $+$ *1*) $*$ *measure N* (*space N*)))
by *simp*
also have ... $\le$ (*1* / *3*) $*$ *e*
proof $-$
have *measure N* (*space N*) $=$ ($\int$ *x*. *1* ∂N)
by *simp*
also have ... $=$ φn *n* ($\lambda x \in$*topspace X*. *1*)
by(*intro N'*(*3*)[*symmetric*]) *simp*
also have ... $\le$ *r*
using *N*(*1*)[*OF n*] **by**(*auto simp*: Φ-*def B-def*)
finally have (*1* / (*r* $+$ *1*) $*$ *measure N* (*space N*)) $\le$ *1*
using *r* **by** *simp*
thus *?thesis*
unfolding *mult-le-cancel-left2* **using** *e* **by** *auto*
qed
finally show *?thesis* **.**
qed
show *?thesis*
using *1 2 3* **by** *simp*
qed
also have ... $=$ *e*
by *simp*
finally show *norm* (φn *n f* $-$ φ *f*) $<$ *e* **.**
qed
qed
thus *limitin euclideanreal* (λn. φn *n f*) (φ *f*) *sequentially*

by *simp*
qed
next
show ⋀*x*. *x* ∈ *topspace d.mtopology* ⟹ *x* ∈ *extensional* (*mspace* (*cfunspace X euclidean-metric*))
unfolding *d.topspace-mtopology* **by** (*auto simp*: Φ-*def prod-space-def extensional-def simp del*: *mspace-cfunspace*)
qed (*simp*, *auto simp*: Φ-*def*)

thus ⋀*S*. *openin* Φ *S* ⟹ *openin d.mtopology S*
by (*metis* Φ-*def d.topspace-mtopology topology-finer-continuous-id*)
next
have *continuous-map* Φ *d.mtopology id*
unfolding *d.continuous-map-to-metric id-apply*
proof *safe*
fix φ **and** *e*::*real*
assume φ: φ ∈ *topspace* Φ **and** *e*: *0* < *e*
then obtain *N* **where** *N*: (*1* / *2*)^*N* < *e* / *2*
by (*meson half-gt-zero-iff one-less-numeral-iff reals-power-lt-ex semiring-norm*(*76*))
define *e′* **where** *e′* ≡ *e* / *2* − (*1* / *2*)^*N*
have *e′*: *0* < *e′*
using *N* **by**(*auto simp*: *e′-def*)
define *U′* **where** *U′* ≡ Π_E *f*∈*mspace* (*cfunspace X euclidean-metric*).
if ∃ *n*<*N*. *f* = *gn n then* {φ (λ*x*∈*topspace X*. *f x*) − *e′*<..<φ (λ*x*∈*topspace X*. *f x*) + *e′*} *else UNIV*
define *U* **where** *U* ≡ *U′* ∩ *B*
show ∃ *U*. *openin* Φ *U* ∧ φ ∈ *U* ∧ (∀ *y*∈*U*. *y* ∈ *d.mball* φ *e*)
proof(*safe intro*!: *exI*[**where** *x*=*U*])
show *openin* Φ *U*
unfolding Φ-*def openin-subtopology U-def*
proof(*safe intro*!: *exI*[**where** *x*=*U′*])
show *openin prod-space U′*
unfolding *prod-space-def U′-def openin-PiE-gen*
by (*auto simp*: *Let-def*)
qed
next
show φ ∈ *U*
unfolding *U-def U′-def*
proof *safe*
fix *f* :: *′a* ⇒ *real*
assume *f*: *f* ∈ *mspace* (*cfunspace X euclidean-metric*)
then show φ *f* ∈ (*if* ∃ *n*<*N*. *f* = *gn n*
then {φ (*restrict f* (*topspace X*)) − *e′*<..<φ (*restrict f* (*topspace X*)) + *e′*}
else UNIV)
using *e′* **by**(*auto simp*: *Let-def gn-ext*)
qed(*use* φ Φ-*def prod-space-def* **in** *auto*)
next

fix ψ
assume ψ:$\psi \in U$
then have $\psi 2$: $\psi \in$ *topspace* Φ
using *topspace-subtopology-subset*[*OF B*] **by**(*auto simp*: *U-def* Φ*-def*)
have ψ*-le*: $|\varphi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x) - \psi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x)| <$ e' **if** n: $n < N$ **for** n
proof −
have $\psi \in (\Pi_E\ f{\in}mspace\ (cfunspace\ X\ euclidean\text{-}metric).$
if $\exists\, n{<}N.\ f = gn\ n$
then $\{\varphi\ (restrict\ f\ (topspace\ X)) - e'{<}..{<}\varphi\ (restrict\ f\ (topspace\ X)) + e'\}$
else UNIV)
using ψ **by**(*auto simp*: *U-def U'-def*)
from *PiE-mem*[*OF this gn-in*(*2*)[*of n*]]
have $\psi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x) \in (if\ \exists\, m{<}N.\ gn\ n = gn\ m$
then $\{\varphi\ (restrict\ (gn\ n)\ (topspace\ X)) - e'{<}..{<}\varphi\ (restrict\ (gn\ n)\ (topspace\ X)) + e'\}$
else UNIV)
by(*simp add*: *gn-ext*)
thus *?thesis*
by (*metis abs-diff-less-iff diff-less-eq greaterThanLessThan-iff n*)
qed
have $d\ \varphi\ \psi < e$
proof −
have $d\ \varphi\ \psi = (\sum n.\ (1\ /\ 2)\ \hat{}\ (n + N) * mdist\ (capped\text{-}metric\ 1\ euclidean\text{-}metric)$
$(\varphi\ (\lambda x{\in}topspace\ X.\ gn\ (n + N)\ x))$
$(\psi\ (\lambda x{\in}topspace\ X.\ gn\ (n + N)\ x)))$
$+ (\sum n{<}N.\ (1\ /\ 2)\ \hat{}\ n * mdist\ (capped\text{-}metric\ 1\ euclidean\text{-}metric)$
$(\varphi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x))$
$(\psi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x)))$
unfolding *d-def* **by**(*rule suminf-split-initial-segment d-def*) *simp*
also have ... $\le (\sum n.\ (1\ /\ 2)\ \hat{}\ (n + N))$
$+ (\sum n{<}N.\ (1\ /\ 2)\ \hat{}\ n * mdist\ (capped\text{-}metric\ 1\ euclidean\text{-}metric)$
$(\varphi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x))$
$(\psi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x)))$
by(*auto intro*!: *suminf-le mdist-capped summable-ignore-initial-segment*[**where** *k*=*N*])
also have ... $= (1\ /\ 2)\hat{}N * 2$
$+ (\sum n{<}N.\ (1\ /\ 2)\ \hat{}\ n * mdist\ (capped\text{-}metric\ 1\ euclidean\text{-}metric)$
$(\varphi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x))$
$(\psi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x)))$
using *nsum-of-r'*[**where** *r*=*1/2* **and** *K*=*1* **and** *k*=*N*,*simplified*] **by** *simp*
also have ... $\le (1\ /\ 2)\hat{}N * 2$
$+ (\sum n{<}N.\ (1\ /\ 2)\ \hat{}\ n * |\varphi\ (\lambda x{\in}topspace\ X.\ gn\ n\ x) - \psi$

```
(λx∈topspace X. gn n x)|)
            by(auto intro!: sum-mono mdist-capped-le[where m=euclidean-metric
:: real metric,simplified,simplified dist-real-def])
          also have ... ≤ (1 / 2)^N ∗ 2 + (∑n<N. (1 / 2) ^ n ∗ e′)
            using ψ-le by(fastforce intro!: sum-mono)
          also have ... < (1 / 2)^N ∗ 2 + (∑n<Suc N. (1 / 2) ^ n ∗ e′)
            using e′ by(auto intro!: sum-strict-mono2)
          also have ... ≤ (1 / 2)^N ∗ 2 + (∑n. (1 / 2) ^ n ∗ e′)
              using e′ by(auto intro!: sum-le-suminf summable-mult2 simp del:
sum.lessThan-Suc)
          also have ... = (1 / 2)^N ∗ 2 + (∑n. (1 / 2) ^ n) ∗ e′
            by(auto intro!: suminf-mult2[symmetric])
          also have ... = (1 / 2)^N ∗ 2 + 2 ∗ e′
            by(auto simp: suminf-geometric)
          also have ... = e
            by(auto simp: e′-def)
          finally show ?thesis .
        qed
        with φ ψ2 show ψ ∈ d.mball φ e
          by simp
      qed
    qed
    thus ⋀S. openin d.mtopology S ⟹ openin Φ S
      by (metis d.topspace-mtopology topology-finer-continuous-id)
  qed
  thus ?thesis
    using d.metrizable-space-mtopology by presburger
next
  case r:2
  have False if h:φ ∈ B for φ
  proof −
    have 1: φ (λx∈topspace X. 1) ≤ r
      using h by(auto simp: B-def)
    have 2: φ (λx∈topspace X. 1) ≥ 0
        using h by(auto simp: B-def pos-lin-functional-on-CX-compact-pos[OF -
compact])
    from 1 2 r show False by linarith
  qed
  hence B = {}
    by auto
  thus ?thesis
    by(auto simp: Φ-def)
qed
qed
```

2.2 Alaoglu's Theorem

According to Alaoglu's theorem, $\{\varphi \in C(X)^* \mid \|\varphi\| \le r\}$ is compact. We show that $\Phi = \{\varphi \in C(X)^* \mid \|\varphi\| \le r \land \varphi \text{ is positive}\}$ is compact. Note that

$\|\varphi\| = \varphi(1)$ when $\varphi \in C(X)^*$ is positive.

theorem *Alaoglu-theorem-real-functional*:
fixes *X* :: *'a topology* **and** *r* :: *real*
defines *prod-space* ≡ *powertop-real* (*mspace* (*cfunspace X euclidean-metric*))
defines *B* ≡ {*φ*∈*topspace prod-space*. *φ* (*λx*∈*topspace X*. *1*) ≤ *r* ∧ *positive-linear-functional-on-CX X φ*}
assumes *compact*: *compact-space X* **and** *ne*: *topspace X* ≠ {}
shows *compactin prod-space B*
proof −
consider *r* ≥ *0* | *r* < *0*
by *linarith*
then show *?thesis*
proof *cases*
assume *rpos*:*r* ≥ *0*
have *continuous-map-compact-space-bounded*: ⋀*f*. *continuous-map X euclidean-real f* ⟹ *bounded* (*f* ‘ *topspace X*)
by (*meson compact compact-imp-bounded compact-space-def compactin-euclidean-iff image-compactin*)
have *1*: *compactin prod-space*
(Π$_E$ *f*∈*mspace* (*cfunspace X euclidean-metric*). {− *r* ∗ (⨆*x*∈*topspace X*. |*f x*|).. *r* ∗ (⨆*x*∈*topspace X*. |*f x*|)})
by(*auto simp*: *prod-space-def compactin-PiE*)
have *2*: *B* ⊆ (Π$_E$ *f*∈*mspace* (*cfunspace X euclidean-metric*). {− *r* ∗ (⨆*x*∈*topspace X*. |*f x*|).. *r* ∗ (⨆*x*∈*topspace X*. |*f x*|)})
proof *safe*
fix *φ* **and** *f* :: *'a* ⇒ *real*
assume *h*:*φ* ∈ *B f* ∈ *mspace* (*cfunspace X euclidean-metric*)
then have *f*: *continuous-map X euclideanreal f f* ∈ *topspace X* →$_E$ *UNIV*
by (*auto simp*: *extensional-def*)
have *plf*:*positive-linear-functional-on-CX X φ*
using *h*(*1*) **by**(*auto simp*: *B-def*)
note *φ* = *pos-lin-functional-on-CX-compact-lin*[*OF plf compact*]
pos-lin-functional-on-CX-compact-pos[*OF plf compact*]
note *φ-mono* = *pos-lin-functional-on-CX-compact-mono*[*OF plf compact*]
note *φ-neg* = *pos-lin-functional-on-CX-compact-uminus*[*OF plf compact f*(*1*),*symmetric*]
obtain *K* **where** *K*: ⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *K*
using *h*(*2*) *bounded-real* **by** *auto*
have *f-Sup*: ⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ (⨆*x*∈*topspace X*. |*f x*|)
by(*auto intro*!: *cSup-upper bdd-aboveI*[**where** *M*=*B*] *K*)
hence *f-Sup-nonneg*: (⨆*x*∈*topspace X*. |*f x*|) ≥ *0*
using *ne* **by** *fastforce*
have |*φ f*| = |*φ* (*λx*∈*topspace X*. *f x*)|
using *f*(*2*) **by** *fastforce*
also have ... ≤ *φ* (*λx*∈*topspace X*. |*f x*|)
using *φ-mono*[*OF* - *f*(*1*) *continuous-map-norm*[*OF f*(*1*),*simplified*]]
φ(*3*)[*OF continuous-map-norm*[*OF f*(*1*),*simplified*]]
φ-mono[*OF* - *continuous-map-minus*[*OF f*(*1*)] *continuous-map-norm*[*OF f*(*1*),*simplified*]]

by(*cases* φ (*restrict f* (*topspace X*)) $\geq$ *0*) (*auto simp*: *φ-neg*)
also have ... $\leq$ φ ($\lambda x \in$*topspace X*. ($\bigsqcup x \in$*topspace X*. $|f\ x|$) $*$ *1*)
using *continuous-map-norm*[**where** *′b=real*] *f*(*1*) *f-Sup*
by(*intro φ-mono*) *auto*
also have ... = ($\bigsqcup x \in$*topspace X*. $|f\ x|$) $*$ φ ($\lambda x \in$*topspace X*. *1*)
by(*intro* φ) *simp*
also have ... $\leq$ r $*$ ($\bigsqcup x \in$*topspace X*. $|f\ x|$)
using *h*(*1*) *f-Sup-nonneg* **by**(*auto simp*: *B-def mult.commute mult-right-mono*)
finally show φ f $\in$ {$-$ r $*$ ($\bigsqcup x \in$*topspace X*. $|f\ x|$).. r $*$ ($\bigsqcup x \in$*topspace X*. $|f$ $x|$)}
by *auto*
qed (*auto simp*: *prod-space-def B-def*)
have *3*: *closedin prod-space B*
proof(*rule closedin-limitin*)
fix *φn φ*
assume *h*:$\bigwedge U$. $\varphi \in U \Longrightarrow$ *openin prod-space U* $\Longrightarrow$ *φn U* $\neq$ φ
$\bigwedge U$. $\varphi \in U \Longrightarrow$ *openin prod-space U* $\Longrightarrow$ *φn U* $\in$ *B*
limitin prod-space φn φ (*nhdsin-sets prod-space φ*)
then have *xnx*:$\varphi \in$ *extensional* (*mspace* (*cfunspace X euclidean-metric*))
($\forall_F$ *U in nhdsin-sets prod-space φ*. *φn U* $\in$ *topspace prod-space*)
$\bigwedge f$. $f \in$*mspace* (*cfunspace X euclidean-metric*) $\Longrightarrow$ *limitin euclideanreal* (λc. *φn c f*) (φ *f*) (*nhdsin-sets prod-space φ*)
by(*auto simp*: *limitin-componentwise prod-space-def*)
have *φ-top*:$\varphi \in$ *topspace prod-space*
by (*meson h*(*3*) *limitin-topspace*)
show $\varphi \in B$
unfolding *B-def*
proof *safe*
have *limit*: *limitin euclideanreal* (λc. *φn c* ($\lambda x \in$*topspace X*. *1*)) (φ ($\lambda x \in$*topspace X*. *1*)) (*nhdsin-sets prod-space φ*)
by(*rule xnx*(*3*)) (*auto simp*: *bounded-iff*)
show φ ($\lambda x \in$*topspace X*. *1*) $\leq r$
using *h*(*2*)
by(*auto intro*!: *tendsto-upperbound*[*OF limit*[*simplified*] - *nhdsin-sets-bot*[*OF φ-top*]]
eventually-nhdsin-setsI[*OF φ-top*] *simp*: *B-def*)
next
show *positive-linear-functional-on-CX X φ*
unfolding *positive-linear-functional-on-CX-compact*[*OF compact*]
proof *safe*
fix *c f*
assume *f*: *continuous-map X euclideanreal f*
then have *f′*:($\lambda x \in$*topspace X*. $c * f\ x$) $\in$ *mspace* (*cfunspace X euclidean-metric*)
($\lambda x \in$*topspace X*. *f x*) $\in$ *mspace* (*cfunspace X euclidean-metric*)
by(*auto simp*: *intro*!: *continuous-map-compact-space-bounded continuous-map-real-mult-left*)
have *tends1*:((λU. c $*$ *φn U* ($\lambda x \in$*topspace X*. *f x*)) $\longrightarrow$ φ ($\lambda x \in$*topspace X*. $c * f\ x$)) (*nhdsin-sets prod-space φ*)

using *B-def f h*(*2*) **by**(*fastforce intro*!: *tendsto-cong*[*THEN iffD1*,*OF* - *xnx*(*3*)[*OF f*′(*1*),*simplified*]]
eventually-nhdsin-setsI[*OF* φ*-top*] *pos-lin-functional-on-CX-compact-lin*[*OF* - *compact f*])
show φ (λ*x*∈*topspace X*. *c* ∗ *f x*) = *c* ∗ φ (λ*x*∈*topspace X*. *f x*)
by(*rule tendsto-unique*[*OF nhdsin-sets-bot*[*OF* φ*-top*] *tends1 tendsto-mult-left*[*OF xnx*(*3*)[*OF f*′(*2*),*simplified*]]]])
next
fix *f g*
assume *fg*:*continuous-map X euclideanreal f continuous-map X euclideanreal g*
then have *fg*′: (λ*x*∈*topspace X*. *f x*) ∈ *mspace* (*cfunspace X euclidean-metric*)
(λ*x*∈*topspace X*. *g x*) ∈ *mspace* (*cfunspace X euclidean-metric*)
(λ*x*∈*topspace X*. *f x* + *g x*) ∈ *mspace* (*cfunspace X euclidean-metric*)
by(*auto intro*!: *continuous-map-compact-space-bounded continuous-map-add*)
have ((λ*c*. φ*n c* (λ*x*∈*topspace X*. *f x*) + φ*n c* (λ*x*∈*topspace X*. *g x*))
⟶ φ (λ*x*∈*topspace X*. *f x* + *g x*)) (*nhdsin-sets prod-space* φ)
using *B-def fg h*(*2*)
by(*fastforce intro*!: *tendsto-cong*[*THEN iffD1*,*OF* - *xnx*(*3*)[*OF fg*′(*3*),*simplified*]]
eventually-nhdsin-setsI[*OF* φ*-top*] *pos-lin-functional-on-CX-compact-lin*[*OF* - *compact*])
moreover have ((λ*c*. φ*n c* (λ*x*∈*topspace X*. *f x*) + φ*n c* (λ*x*∈*topspace X*. *g x*))
⟶ φ (λ*x*∈*topspace X*. *f x*) + φ (λ*x*∈*topspace X*. *g x*)) (*nhdsin-sets prod-space* φ)
using *xnx fg*′ **by**(*auto intro*!: *tendsto-add*)
ultimately show φ (λ*x*∈*topspace X*. *f x* + *g x*) = φ (λ*x*∈*topspace X*. *f x*) + φ (λ*x*∈*topspace X*. *g x*)
by(*rule tendsto-unique*[*OF nhdsin-sets-bot*[*OF* φ*-top*]])
next
fix *f*
assume *f*:*continuous-map X euclideanreal f* ∀ *x*∈*topspace X*. *0* ≤ *f x*
then have *1*:(λ*x*∈*topspace X*. *f x*) ∈ *mspace* (*cfunspace X euclidean-metric*)
by(*auto intro*!: *continuous-map-compact-space-bounded*)
from *f h*(*2*) **show** *0* ≤ φ (λ*x*∈*topspace X*. *f x*)
by(*auto intro*!: *tendsto-lowerbound*[*OF xnx*(*3*)[*OF 1*,*simplified*] - *nhdsin-sets-bot*[*OF* φ*-top*]]
eventually-nhdsin-setsI[*OF* φ*-top*] *simp*: *B-def pos-lin-functional-on-CX-compact-pos*[*OF* - *compact f*(*1*)]])
qed
qed *fact*
qed(*auto simp*: *B-def*)
show *?thesis*
using *1 2 3* **by**(*rule closed-compactin*)
next
assume *r*:*r* < *0*
have *B* = {}
proof *safe*

fix φ
assume *h*:$\varphi \in B$
then have $\bigwedge f$. *continuous-map X euclideanreal f* $\Longrightarrow$ ($\bigwedge x$. *x* $\in$ *topspace X* $\Longrightarrow$ *f x* $\geq$ *0*) $\Longrightarrow$ φ ($\lambda x \in$*topspace X*. *f x*) $\geq$ *0*
by(*auto simp*: *B-def pos-lin-functional-on-CX-compact-pos*[*OF - compact*])
from *this*[*of* λx. *1*] *h r* **show** $\varphi \in \{\}$
by(*auto simp*: *B-def*)
qed
thus *compactin prod-space B*
by *blast*
qed
qed

theorem *Alaoglu-theorem-real-functional-seq*:
fixes *X* :: *'a topology* **and** *r* :: *real*
defines *prod-space* ≡ *powertop-real* (*mspace* (*cfunspace X euclidean-metric*))
defines *B* ≡ {$\varphi \in$*topspace prod-space*. φ ($\lambda x \in$*topspace X*. *1*) $\leq$ *r* $\wedge$ *positive-linear-functional-on-CX X* φ}
assumes *compact*:*compact-space X* **and** *ne*: *topspace X* $\neq$ {} **and** *met*: *metrizable-space X*
shows *seq-compactin prod-space B*
proof −
have *compactin prod-space B*
using *Alaoglu-theorem-real-functional*[*OF compact ne*] **by**(*auto simp*: *B-def prod-space-def*)
hence *compact-space* (*subtopology prod-space B*)
using *compact-space-subtopology* **by** *blast*
hence *seq-compact-space* (*subtopology prod-space B*)
unfolding *B-def prod-space-def*
using *metrizable-seq-compact-space-iff-compact-space*[*OF metrizable-functional*[*OF compact met*]]
by *fast*
moreover have *B* $\subseteq$ *topspace prod-space*
by(*auto simp*: *B-def*)
ultimately show *?thesis*
by (*simp add*: *inf.absorb-iff2 seq-compact-space-def seq-compactin-subtopology*)
qed

end

3 General Weak Convergence

theory *General-Weak-Convergence*
imports *Lemmas-Levy-Prokhorov*
Riesz-Representation.Regular-Measure
begin

We formalize the notion of weak convergence and equivalent conditions. The formalization of weak convergence in HOL-Probability is restricted to probability measures on real numbers. Our formalization is generalized to finite measures on any metric spaces.

3.1 Topology of Weak Convegence

definition *weak-conv-topology* :: *'a topology* ⇒ *'a measure topology* **where**
weak-conv-topology X ≡
  topology-generated-by
    (⋃*f*∈{*f*. *continuous-map X euclideanreal f* ∧ (∃ *B*. ∀ *x*∈*topspace X*. |*f x*| ≤ *B*)}.
        Collect (*openin* (*pullback-topology* {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
                                (λ*N*. ∫ *x*. *f x* ∂*N*) *euclideanreal*)))

lemma *topspace-weak-conv-topology*[*simp*]:
topspace (*weak-conv-topology X*) = {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
  unfolding *weak-conv-topology-def openin-pullback-topology*
  by(*auto intro*!: *exI*[**where** *x*=λ*x*. *1*] *exI*[**where** *x*=*1*]) *blast*

lemma *openin-weak-conv-topology-base*:
  assumes *f*:*continuous-map X euclideanreal f* **and** *B*:⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*
    and *U*:*open U*
  shows *openin* (*weak-conv-topology X*) ((λ*N*. ∫ *x*. *f x* ∂*N*) −‘ *U*
                                ∩ {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*})
  using *assms*
  by(*fastforce simp*: *weak-conv-topology-def openin-topology-generated-by-iff openin-pullback-topology*
      intro!: *Basis*)

lemma *continuous-map-weak-conv-topology*:
  assumes *f*:*continuous-map X euclideanreal f* **and** *B*:⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*
  shows *continuous-map* (*weak-conv-topology X*) *euclideanreal* (λ*N*. ∫ *x*. *f x* ∂*N*)
  using *openin-weak-conv-topology-base*[*OF assms*]
  by(*auto simp*: *continuous-map-def Collect-conj-eq Int-commute Int-left-commute vimage-def*)

lemma *weak-conv-topology-minimal*:
  assumes *topspace Y* = {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
    and ⋀*f B*. *continuous-map X euclideanreal f*
              ⟹ (⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*) ⟹ *continuous-map Y euclideanreal* (λ*N*. ∫ *x*. *f x* ∂*N*)
  shows *openin* (*weak-conv-topology X*) *U* ⟹ *openin Y U*
  unfolding *weak-conv-topology-def openin-topology-generated-by-iff*
proof (*induct rule*: *generate-topology-on.induct*)
  case *h*:(*Basis s*)
  then obtain *f B* **where** *f*: *continuous-map X euclidean f* ⋀*x*. *x*∈*topspace X* ⟹ |*f x*| ≤ *B*

openin (*pullback-topology* {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
(λ*N*. ∫ *x*. *f x* ∂*N*) *euclideanreal*) *s*
by *blast*
then obtain *u* **where** *u*:
open u s = (λ*N*. ∫ *x*. *f x* ∂*N*) −‘*u* ∩ {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
unfolding *openin-pullback-topology* **by** *auto*
with *assms*(*2*)[*OF f*(*1*,*2*)]
show *?case*
using *assms*(*1*) *continuous-map-open* **by** *fastforce*
qed *auto*

lemma *weak-conv-topology-continuous-map-integral*:
assumes *continuous-map X euclideanreal f* ⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*
shows *continuous-map* (*weak-conv-topology X*) *euclideanreal* (λ*N*. ∫ *x*. *f x* ∂*N*)
unfolding *continuous-map*
proof *safe*
fix *U*
assume *openin euclideanreal U*
then show *openin* (*weak-conv-topology X*) {*N* ∈ *topspace* (*weak-conv-topology X*). (∫ *x*. *f x* ∂*N*) ∈ *U*}
unfolding *weak-conv-topology-def openin-topology-generated-by-iff* **using** *assms*
by(*auto intro*!: *Basis exI*[**where** *x*=*U*] *exI*[**where** *x*=*f*] *exI*[**where** *x*=*B*] *simp*: *openin-pullback-topology*) *blast*
qed *simp*

3.2 Weak Convergence

abbreviation *weak-conv-on* :: (′*a* ⇒ ′*b measure*) ⇒ ′*b measure* ⇒ ′*a filter* ⇒ ′*b topology* ⇒ *bool*
(′((-)/ $\Rightarrow_{WC}$ (-)′) (-)/ *on* (-) [*56*, *55*] *55*) **where**
weak-conv-on Ni N F X ≡ *limitin* (*weak-conv-topology X*) *Ni N F*

abbreviation *weak-conv-on-seq* :: (*nat* ⇒ ′*b measure*) ⇒ ′*b measure* ⇒ ′*b topology* ⇒ *bool*
(′((-)/ $\Rightarrow_{WC}$ (-)′) *on* (-) [*56*, *55*] *55*) **where**
weak-conv-on-seq Ni N X ≡ *weak-conv-on Ni N sequentially X*

3.3 Limit in Topology of Weak Convegence

lemma *weak-conv-on-def*:
weak-conv-on Ni N F X ⟷
($\forall_F$ *i in F*. *sets* (*Ni i*) = *sets* (*borel-of X*) ∧ *finite-measure* (*Ni i*)) ∧ *sets N* = *sets* (*borel-of X*)
∧ *finite-measure N*
∧ (∀*f*. *continuous-map X euclideanreal f* ⟶ (∃ *B*. ∀ *x*∈*topspace X*. |*f x*| ≤ *B*)
⟶ ((λ*i*. ∫ *x*. *f x* ∂*Ni i*) ⟶ (∫ *x*. *f x* ∂*N*)) *F*)
proof *safe*
assume *h*:*weak-conv-on Ni N F X*

then have *1*:*sets N* = *sets* (*borel-of X*) *finite-measure N*
  using *limitin-topspace* **by** *fastforce*+
then show $\bigwedge x.\ x \in$ *sets N* $\Longrightarrow x \in$ *sets* (*borel-of X*) $\bigwedge x.\ x \in$ *sets* (*borel-of X*) $\Longrightarrow x \in$ *sets N*
  finite-measure N
  by *auto*
show *2*:$\forall_F$ *i in F*. *sets* (*Ni i*) = *sets* (*borel-of X*) ∧ *finite-measure* (*Ni i*)
  using *h* **by**(*cases F* = ⊥) (*auto simp*: *limitin-def*)
fix *f B*
assume *f*:*continuous-map X euclideanreal f* **and** *B*:∀ *x*∈*topspace X*. |*f x*| ≤ *B*
show ((λ*n*. ∫ *x*. *f x* ∂*Ni n*) ⟶ (∫ *x*. *f x* ∂*N*)) *F*
  unfolding *tendsto-iff*
proof *safe*
  fix *r* :: *real*
  assume [*arith*]:*r* > *0*
  then have *openin*
      (*weak-conv-topology X*)
      ((λ*N*. ∫ *x*. *f x* ∂*N*) −‘ (*ball* (∫ *x*. *f x* ∂*N*) *r*)
       ∩ {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}) (**is** *openin* - *?U*)
    using *f B* **by**(*auto intro*!: *openin-weak-conv-topology-base*)
  moreover have *N* ∈ *?U*
    using *h* **by** (*simp add*: *1*)
  ultimately have *NnU*:$\forall_F$ *n in F*. *Ni n* ∈ *?U*
    using *h limitinD* **by** *fastforce*
  show $\forall_F$ *n in F*. *dist* (∫ *x*. *f x* ∂*Ni n*) (∫ *x*. *f x* ∂*N*) < *r*
   by(*auto intro*!: *eventuallyI*[*THEN eventually-mp*[*OF* - *NnU*]] *simp*: *dist-real-def*)
qed
next
assume *h*: $\forall_F$ *i in F*. *sets* (*Ni i*) = *sets* (*borel-of X*) ∧ *finite-measure* (*Ni i*)
     sets N = *sets* (*borel-of X*)
     finite-measure N
     ∀ *f*. *continuous-map X euclideanreal f* ⟶ (∃ *B*. ∀ *x*∈*topspace X*. |*f x*| ≤ *B*)
       ⟶ ((λ*n*. ∫ *x*. *f x* ∂*Ni n*) ⟶ (∫ *x*. *f x* ∂*N*)) *F*
show (*Ni* $\Rightarrow_{WC}$ *N*) *F on X*
  unfolding *limitin-def*
proof *safe*
  show *N* ∈ *topspace* (*weak-conv-topology X*)
    using *h* **by** *auto*
  fix *U*
  assume *h'*:*openin* (*weak-conv-topology X*) *U N* ∈ *U*
  show $\forall_F$ *x in F*. *Ni x* ∈ *U*
    using *h'*[*simplified weak-conv-topology-def openin-topology-generated-by-iff*]
  proof *induction*
    case *Empty*
    then show *?case*
      by *simp*
  next

case (*Int a b*)
then show *?case*
by (*simp add*: *eventually-conj-iff*)
next
case (*UN K*)
then show *?case*
using *UnionI eventually-mono* **by** *fastforce*
next
case *s*:(*Basis s*)
then obtain *f* **where** *f*: *continuous-map X euclidean f* $\exists B.\ \forall x{\in}topspace\ X.$ $|f\ x| \leq B$
openin (*pullback-topology* $\{N.$ *sets* $N =$ *sets* (*borel-of* X) $\wedge$
finite-measure $N\}$ ($\lambda N.\ \int x.\ f\ x\ \partial N$) *euclideanreal*) *s*
by *blast*
then obtain *u* **where** *u*:
open u s = ($\lambda N.\ \int x.\ f\ x\ \partial N$) $-$'$u \cap \{N.$ *sets* $N =$ *sets* (*borel-of* X) $\wedge$ *finite-measure* $N\}$
unfolding *openin-pullback-topology* **by** *auto*
have ($\int x.\ f\ x\ \partial N$) $\in u$
using *u s* **by** *blast*
moreover have (($\lambda n.\ \int x.\ f\ x\ \partial Ni\ n$) $\longrightarrow$ ($\int x.\ f\ x\ \partial N$)) *F*
using *f h* **by** *blast*
ultimately have *1*:$\forall_F$ *n in F*. ($\int x.\ f\ x\ \partial(Ni\ n)$) $\in u$
by (*simp add*: *tendsto-def u*(*1*))
show *?case*
by(*auto intro*!: *eventuallyI*[*THEN eventually-mp*[*OF - eventually-conj*[*OF 1 h*(*1*)]]] *simp*: *u*(*2*))
qed
qed
qed

lemma *weak-conv-on-def'*:
assumes $\bigwedge i.$ *sets* (*Ni i*) = *sets* (*borel-of X*) **and** $\bigwedge i.$ *finite-measure* (*Ni i*)
and *sets N* = *sets* (*borel-of X*) **and** *finite-measure N*
shows *weak-conv-on Ni N F X*
$\longleftrightarrow$ ($\forall f.$ *continuous-map X euclideanreal f* $\longrightarrow$ ($\exists B.\ \forall x{\in}topspace\ X.\ |f\ x| \leq B$)
$\longrightarrow$ (($\lambda i.\ \int x.\ f\ x\ \partial Ni\ i$) $\longrightarrow$ ($\int x.\ f\ x\ \partial N$)) *F*)
using *assms* **by**(*auto simp*: *weak-conv-on-def*)

lemmas *weak-conv-seq-def* = *weak-conv-on-def*[**where** *F*=*sequentially*]

lemma *weak-conv-on-const*:
($\bigwedge i.$ *Ni i* = *N*) $\Longrightarrow$ *sets N* = *sets* (*borel-of X*)
$\Longrightarrow$ *finite-measure N* $\Longrightarrow$ *weak-conv-on Ni N F X*
by(*auto simp*: *weak-conv-on-def*)

lemmas *weak-conv-on-seq-const* = *weak-conv-on-const*[**where** *F*=*sequentially*]

context *Metric-space*
begin

abbreviation *mweak-conv* ≡ (*λNi N F*. *weak-conv-on Ni N F mtopology*)
abbreviation *mweak-conv-seq* ≡ *λNi N*. *mweak-conv Ni N sequentially*

lemmas *mweak-conv-def* = *weak-conv-on-def*[**where** *X*=*mtopology*,*simplified*]
lemmas *mweak-conv-seq-def* = *weak-conv-seq-def*[**where** *X*=*mtopology*,*simplified*]

end

3.4 The Portmanteau Theorem

locale *mweak-conv-fin* = *Metric-space* +
  fixes *Ni* :: *'b* ⇒ *'a measure* **and** *N* :: *'a measure* **and** *F*
  assumes *sets-Ni*:$\forall_F$ *i in F*. *sets* (*Ni i*) = *sets* (*borel-of mtopology*)
    and *sets-N*[*measurable-cong*]: *sets N* = *sets* (*borel-of mtopology*)
    and *finite-measure-Ni*: $\forall_F$ *i in F*. *finite-measure* (*Ni i*)
    and *finite-measure-N*: *finite-measure N*
begin

interpretation *N*: *finite-measure N*
  by(*simp add*: *finite-measure-N*)

lemma *space-N*: *space N* = *M*
  using *sets-eq-imp-space-eq*[*OF sets-N*] **by**(*auto simp*: *space-borel-of*)

lemma *space-Ni*: $\forall_F$ *i in F*. *space* (*Ni i*) = *M*
 by(*rule eventually-mp*[*OF - sets-Ni*]) (*auto simp*: *space-borel-of cong*: *sets-eq-imp-space-eq*)

lemma *eventually-Ni*: $\forall_F$ *i in F*. *space* (*Ni i*) = *M* ∧ *sets* (*Ni i*) = *sets* (*borel-of mtopology*) ∧ *finite-measure* (*Ni i*)
  by(*intro eventually-conj space-Ni sets-Ni finite-measure-Ni*)

lemma *measure-converge-bounded'*:
  assumes ((*λn*. *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
  obtains *K* **where** ⋀*A*. $\forall_F$ *x in F*. *measure* (*Ni x*) *A* ≤ *K* ⋀*A*. *measure N A* ≤ *K*
proof −
  have *measure N A* ≤ *measure N M* + *1* **for** *A*
    using *N*.*bounded-measure*[*of A*] **by**(*simp add*: *space-N*)
  moreover have $\forall_F$ *x in F*. *measure* (*Ni x*) *A* ≤ *measure N M* + *1* **for** *A*
   proof(*rule eventuallyI*[*THEN eventually-mp*[*OF - eventually-conj*[*OF eventually-Ni tendstoD*[*OF assms*,*of 1*]]]])
    fix *x*
     show (*space* (*Ni x*) = *M* ∧ *sets* (*Ni x*) = *sets* (*borel-of mtopology*) ∧ *finite-measure* (*Ni x*)) ∧
        dist (*measure* (*Ni x*) *M*) (*measure N M*) < *1* ⟶ *measure* (*Ni x*) *A* ≤ *measure N M* + *1*

using *finite-measure.bounded-measure*[*of Ni x A*]
by(*auto intro*!: *eventuallyI*[*THEN eventually-mp*[*OF - tendstoD*[*OF assms*,*of 1*]]] *simp*: *dist-real-def*)
qed *simp*
ultimately show *?thesis*
using *that* **by** *blast*
qed

lemma
assumes $F \neq \bot$ $\forall_F$ *x in F*. *measure* (*Ni x*) $A \leq K$ *measure N* $A \leq K$
shows *Liminf-measure-bounded*: *Liminf F* (λi. *measure* (*Ni i*) *A*) $< \infty$ $0 \leq$ *Liminf F* (λi. *measure* (*Ni i*) *A*)
and *Limsup-measure-bounded*: *Limsup F* (λi. *measure* (*Ni i*) *A*) $< \infty$ $0 \leq$ *Limsup F* (λi. *measure* (*Ni i*) *A*)
proof –
have *Liminf F* (λi. *measure* (*Ni i*) *A*) $\leq K$ *Limsup F* (λi. *measure* (*Ni i*) *A*) $\leq K$
using *assms* **by**(*auto intro*!: *Liminf-le Limsup-bounded*)
thus *Liminf F* (λi. *measure* (*Ni i*) *A*) $< \infty$ *Limsup F* (λi. *measure* (*Ni i*) *A*) $< \infty$
by *auto*
show $0 \leq$ *Liminf F* (λi. *measure* (*Ni i*) *A*) $0 \leq$ *Limsup F* (λi. *measure* (*Ni i*) *A*)
by(*auto intro*!: *le-Limsup Liminf-bounded assms*)
qed

lemma *mweak-conv1*:
fixes *f*:: *'a* $\Rightarrow$ *real*
assumes *mweak-conv Ni N F*
and *uniformly-continuous-map Self euclidean-metric f*
shows ($\exists B. \forall x \in M. |f\ x| \leq B$) $\Longrightarrow$ ((λn. *integral*^{L} (*Ni n*) *f*) $\longrightarrow$ *integral*^{L} *N f*) *F*
using *uniformly-continuous-imp-continuous-map*[*OF assms*(*2*)] *assms*(*1*) **by**(*auto simp*: *mweak-conv-def mtopology-of-def*)

lemma *mweak-conv2*:
assumes $\bigwedge f$:: *'a* $\Rightarrow$ *real*. *uniformly-continuous-map Self euclidean-metric f* $\Longrightarrow$ ($\exists B. \forall x \in M. |f\ x| \leq B$)
$\Longrightarrow$ ((λn. *integral*^{L} (*Ni n*) *f*) $\longrightarrow$ *integral*^{L} *N f*) *F*
and *closedin mtopology A*
shows *Limsup F* (λx. *ereal* (*measure* (*Ni x*) *A*)) $\leq$ *ereal* (*measure N A*)
proof –
consider $A = \{\}$ | $F = \bot$ | $A \neq \{\}$ $F \neq \bot$
by *blast*
then show *?thesis*
proof *cases*
assume $A = \{\}$
then show *?thesis*
using *Limsup-obtain linorder-not-less* **by** *fastforce*

next
assume *A-ne*: $A \neq \{\}$ **and** *F*: $F \neq \bot$
have *A*[*measurable*]: $A \in$ *sets N* $\forall_F$ *i in F*. $A \in$ *sets* (*Ni i*)
using *borel-of-closed*[*OF assms*(*2*)] **by**(*auto simp*: *sets-N eventually-mp*[*OF - sets-Ni*])
have ((λn. *measure* (*Ni n*) *M*) $\longrightarrow$ *measure N M*) *F*
proof −
have *1*:((λn. *measure* (*Ni n*) (*space* (*Ni n*))) $\longrightarrow$ *measure N M*) *F*
using *assms*(*1*)[*of* λx. *1*] **by**(*auto simp*: *space-N*)
show *?thesis*
by(*rule tendsto-cong*[*THEN iffD1*,*OF eventually-mp*[*OF - space-Ni*] *1*]) *simp*
qed
then obtain *K* **where** *K*: $\bigwedge A$. $\forall_F$ *x in F*. *measure* (*Ni x*) $A \le K$ $\bigwedge A$. *measure N* $A \le K$
using *measure-converge-bounded'* **by** *auto*
define *Um* **where** *Um* $\equiv$ (λm. $\bigcup a \in A$. *mball a* (*1* / *Suc m*))
have *Um-open*: *openin mtopology* (*Um m*) **for** *m*
by(*auto simp*: *Um-def*)
hence *Um-m*[*measurable*]: $\bigwedge m$. *Um m* $\in$ *sets N* $\bigwedge m$. $\forall_F$ *i in F*. *Um m* $\in$ *sets* (*Ni i*)
by(*auto simp*: *sets-N intro*!: *borel-of-open eventually-mono*[*OF sets-Ni*])
have *A-Um*: $A \subseteq$ *Um m* **for** *m*
using *closedin-subset*[*OF assms*(*2*)] **by**(*fastforce simp*: *Um-def*)
have $\exists$*fm*:: - $\Rightarrow$ *real*. ($\forall x$. *fm x* ≥ 0) $\wedge$ ($\forall x$. *fm x* ≤ 1) $\wedge$ ($\forall x \in M$ − *Um m*. *fm x* = *0*) $\wedge$ ($\forall x \in A$. *fm x* = *1*) $\wedge$
uniformly-continuous-map Self euclidean-metric fm **for** *m*
proof −
have *1*: *closedin mtopology* (*M* − *Um m*)
using *Um-open*[*of m*] **by**(*auto simp*: *closedin-def Diff-Diff-Int Int-absorb1*)
have *2*: $A \cap (M - Um\ m) = \{\}$
using *A-Um*[*of m*] **by** *blast*
have *3*: *1* / *Suc m* $\le$ *d x y* **if** $x \in A$ $y \in M$ − *Um m* **for** *x y*
proof(*rule ccontr*)
assume ¬ *1* / *real* (*Suc m*) $\le$ *d x y*
then have *d x y* < *1* / (*1* + *real m*) **by** *simp*
thus *False*
using *that closedin-subset*[*OF assms*(*2*)] **by**(*auto simp*: *Um-def*)
qed
show *?thesis*
by (*metis Urysohn-lemma-uniform*[*of Self*,*simplified mtopology-of-def*,*simplified*,*OF assms*(*2*) *1 2 3*,*simplified*] *Diff-iff*)
qed
then obtain *fm* :: *nat* $\Rightarrow$ - $\Rightarrow$ *real* **where** *fm*: $\bigwedge m\ x$. *fm m x* ≥ 0 $\bigwedge m\ x$. *fm m x* ≤ 1
$\bigwedge m\ x$. $x \in A \Longrightarrow$ *fm m x* = *1* $\bigwedge m\ x$. $x \in M \Longrightarrow x \notin$ *Um m* $\Longrightarrow$ *fm m x* = *0*
$\bigwedge m$. *uniformly-continuous-map Self euclidean-metric* (*fm m*)
by (*metis Diff-iff*)
have *fm-m*[*measurable*]: $\bigwedge m$. $\forall_F$ *i in F*. *fm m* $\in$ *borel-measurable* (*Ni i*) $\bigwedge m$. *fm m* $\in$ *borel-measurable N*

using *continuous-map-measurable*[*OF uniformly-continuous-imp-continuous-map*[*OF fm*(*5*)]]
by(*auto simp*: *borel-of-euclidean mtopology-of-def eventually-mono*[*OF sets-Ni*])
have *int-bounded*:$\forall_F$ *n in F*. ($\int$ *x*. *fm m x* ∂*Ni n*) $\leq$ *K* **for** *m*
proof(*rule eventually-mono*)
show $\forall_F$ *n in F*. *space* (*Ni n*) = *M* $\wedge$ *finite-measure* (*Ni n*) $\wedge$ *fm m* $\in$ *borel-measurable* (*Ni n*) $\wedge$
($\int$ *x*. *fm m x* ∂*Ni n*) $\leq$ ($\int$ *x*. *1* ∂*Ni n*) $\wedge$ ($\int$ *x*. *1* ∂*Ni n*) $\leq$ *K*
proof(*intro eventually-conj*)
show $\forall_F$ *n in F*. ($\int$ *x*. *fm m x* ∂*Ni n*) $\leq$ ($\int$ *x*. *1* ∂*Ni n*)
proof(*rule eventually-mono*)
show $\forall_F$ *n in F*. *space* (*Ni n*) = *M* $\wedge$ *finite-measure* (*Ni n*) $\wedge$ *fm m* $\in$ *borel-measurable* (*Ni n*)
by(*intro eventually-conj space-Ni finite-measure-Ni fm-m*)
show *space* (*Ni n*) = *M* $\wedge$ *finite-measure* (*Ni n*) $\wedge$ *fm m* $\in$ *borel-measurable* (*Ni n*)
$\Longrightarrow$ ($\int$ *x*. *fm m x* ∂*Ni n*) $\leq$ ($\int$ *x*. *1* ∂*Ni n*) **for** *n*
by(*rule integral-mono*, *insert fm*) (*auto intro*!: *finite-measure.integrable-const-bound*[**where** *B*=*1*])
qed
show $\forall_F$ *n in F*. ($\int$ *x*. *1* ∂*Ni n*) $\leq$ *K*
by(*rule eventually-mono*[*OF eventually-conj*[*OF K*(*1*)[*of M*] *space-Ni*]]) *simp*
qed(*auto simp*: *space-Ni finite-measure-Ni fm-m*)
qed *simp*
have *1*: *Limsup F* (λ*n*. *measure* (*Ni n*) *A*) $\leq$ *measure N* (*Um m*) **for** *m*
proof −
have *Limsup F* (λ*n*. *measure* (*Ni n*) *A*) = *Limsup F* (λ*n*. $\int$ *x*. *indicat-real A x* ∂*Ni n*)
by(*intro Limsup-eq*[*OF eventually-mono*[*OF A*(*2*)]]) *simp*
also have ... $\leq$ *Limsup F* (λ*n*. $\int$ *x*. *fm m x* ∂*Ni n*)
proof(*safe intro*!: *eventuallyI*[*THEN Limsup-mono*[*OF eventually-mp*[*OF - eventually-conj*[*OF fm-m*(*1*)[*of m*]
eventually-conj[*OF finite-measure-Ni eventually-conj*[*OF A*(*2*) *int-bounded*[*of m*]]]]]]])
fix *n*
assume *h*:($\int$ *x*. *fm m x* ∂*Ni n*) $\leq$ *K A* $\in$ *sets* (*Ni n*) *finite-measure* (*Ni n*) *fm m* $\in$ *borel-measurable* (*Ni n*)
with *fm* **show** *ereal* ($\int$ *x*. *indicat-real A x* ∂*Ni n*) $\leq$ *ereal* ($\int$ *x*. *fm m x* ∂*Ni n*)
by(*auto intro*!: *Limsup-mono integral-mono finite-measure.integrable-const-bound*[**where** *B*=*1*]
simp del: *Bochner-Integration.integral-indicator*) (*auto simp*: *indicator-def*)
qed
also have ... = ($\int$ *x*. *fm m x* ∂*N*)
using *fm* **by**(*auto intro*!: *lim-imp-Limsup*[*OF F tendsto-ereal*[*OF assms*(*1*)[*OF fm*(*5*)[*of m*]]]] *exI*[**where** *x*=*1*])
also have ... $\leq$ ($\int$ *x*. *indicat-real* (*Um m*) *x* ∂*N*)

unfolding *ereal-less-eq(3)* **by**(*rule integral-mono*, *insert fm(4)[of - m] fm(1,2)*)
(*auto intro*!: *N.integrable-const-bound*[**where** *B=1*],*auto simp*: *indicator-def space-N*)
also have ... = *measure N* (*Um m*)
by *simp*
finally show *?thesis* **.**
qed
have *2*: (λ*n*. *measure N* (*Um n*)) ⟶ *measure N A*
proof −
have [*simp*]: (⋂ (*range Um*)) = *A*
unfolding *Um-def*
by(*rule nbh-Inter-closure-of*[*OF A-ne - - - LIMSEQ-Suc*,*simplified closure-of-closedin*[*OF assms*(*2*)]],
insert sets.sets-into-space[*OF A*(*1*)])
(*auto intro*!: *decseq-SucI simp*: *frac-le space-N lim-1-over-n*)
have [*simp*]: *monotone* (≤) (λ*x y*. *y* ⊆ *x*) *Um*
unfolding *Um-def* **by**(*rule nbh-decseq*) (*auto intro*!: *decseq-SucI simp*: *frac-le*)
have (λ*n*. *measure N* (*Um n*)) ⟶ *measure N* (⋂ (*range Um*))
by(*rule N.finite-Lim-measure-decseq*) *auto*
thus *?thesis* **by** *simp*
qed
show *?thesis*
using *1* **by**(*auto intro*!: *Lim-bounded2*[*OF tendsto-ereal*[*OF 2*]])
qed *simp*
qed

lemma *mweak-conv3*:
assumes ⋀*A*. *closedin mtopology A* ⟹ *Limsup F* (λ*n*. *measure* (*Ni n*) *A*) ≤ *measure N A*
and ((λ*n*. *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
and *openin mtopology U*
shows *measure N U* ≤ *Liminf F* (λ*n*. *measure* (*Ni n*) *U*)
proof(*cases F* = ⊥)
assume *F*: *F* ≠ ⊥
obtain *K* **where** *K*: ⋀*A*. $\forall_F$ *x in F*. *measure* (*Ni x*) *A* ≤ *K* ⋀*A*. *measure N M* ≤ *K*
using *measure-converge-bounded'*[*OF assms*(*2*)] **by** *metis*
have *U*[*measurable*]: *U* ∈ *sets N* $\forall_F$ *i in F*. *U* ∈ *sets* (*Ni i*)
by(*auto simp*: *sets-N borel-of-open assms eventually-mono*[*OF sets-Ni*])
have *ereal* (*measure N U*) = *measure N M* − *measure N* (*M* − *U*)
by(*simp add*: *N.finite-measure-compl*[*simplified space-N*])
also have ... ≤ *measure N M* − *Limsup F* (λ*n*. *measure* (*Ni n*) (*M* − *U*))
using *assms*(*1*)[*OF openin-closedin*[*THEN iffD1*,*OF - assms*(*3*)]] *openin-subset*[*OF assms*(*3*)]
by (*metis ereal-le-real ereal-minus*(*1*) *ereal-minus-mono topspace-mtopology*)
also have ... = *measure N M* + *Liminf F* (λ*n*. − *ereal* (*measure* (*Ni n*) (*M* − *U*)))

by (*metis ereal-Liminf-uminus minus-ereal-def*)
also have ... = *Liminf F* (*λn. measure* (*Ni n*) *M*) + *Liminf F* (*λn.* − *measure* (*Ni n*) (*M* − *U*))
using *tendsto-iff-Liminf-eq-Limsup*[*OF F*, *THEN iffD1*, *OF tendsto-ereal*[*OF assms*(*2*)]] **by** *simp*
also have ... ≤ *Liminf F* (*λn. ereal* (*measure* (*Ni n*) *M*) + *ereal* (− *measure* (*Ni n*) (*M* − *U*)))
by(*rule ereal-Liminf-add-mono*) (*use Liminf-measure-bounded*[*OF F K*] **in** *auto*)
also have ... = *Liminf F* (*λn. measure* (*Ni n*) *U*)
by(*auto intro*!: *Liminf-eq eventually-mono*[*OF eventually-conj*[*OF U*(*2*) *eventually-conj*[*OF space-Ni finite-measure-Ni*]]]
simp: *finite-measure.finite-measure-compl*)
finally show *?thesis* **.**
qed *simp*

lemma *mweak-conv3'*:
assumes $\bigwedge$*U*. *openin mtopology U* ⟹ *measure N U* ≤ *Liminf F* (*λn. measure* (*Ni n*) *U*)
and ((*λn. measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
and *closedin mtopology A*
shows *Limsup F* (*λn. measure* (*Ni n*) *A*) ≤ *measure N A*
proof(*cases F* = ⊥)
assume *F*: *F* ≠ ⊥
have *A*[*measurable*]: *A* ∈ *sets N*∀$_F$ *i in F*. *A* ∈ *sets* (*Ni i*)
by(*auto simp*: *sets-N borel-of-closed assms eventually-mono*[*OF sets-Ni*])
have *Limsup F* (*λn. measure* (*Ni n*) *A*) = *Limsup F* (*λn. ereal* (*measure* (*Ni n*) *M*) + *ereal* (− *measure* (*Ni n*) (*M* − *A*)))
by(*auto intro*!: *Limsup-eq eventually-mono*[*OF eventually-conj*[*OF A*(*2*) *eventually-conj*[*OF space-Ni finite-measure-Ni*]]]
simp: *finite-measure.finite-measure-compl*)
also have ... ≤ *Limsup F* (*λn. measure* (*Ni n*) *M*) + *Limsup F* (*λn.* − *measure* (*Ni n*) (*M* − *A*))
by(*rule ereal-Limsup-add-mono*)
also have ... = *Limsup F* (*λn. measure* (*Ni n*) *M*) + *Limsup F* (*λn.* − *ereal* (*measure* (*Ni n*) (*M* − *A*)))
by *simp*
also have ... = *Limsup F* (*λn. measure* (*Ni n*) *M*) − *Liminf F* (*λn. measure* (*Ni n*) (*M* − *A*))
unfolding *ereal-Limsup-uminus* **using** *minus-ereal-def* **by** *presburger*
also have ... = *measure N M* − *Liminf F* (*λn. measure* (*Ni n*) (*M* − *A*))
by(*simp add*: *lim-imp-Limsup*[*OF F tendsto-ereal*[*OF assms*(*2*)]])
also have ... ≤ *measure N M* − *measure N* (*M* − *A*)
using *assms*(*1*)[*OF openin-diff*[*OF openin-topspace assms*(*3*)]] *closedin-subset*[*OF assms*(*3*)]
by (*metis assms*(*1*,*3*) *ereal-le-real ereal-minus*(*1*) *ereal-minus-mono open-in-mspace openin-diff*)
also have ... = *measure N A*
by(*simp add*: *N.finite-measure-compl*[*simplified space-N*])
finally show *?thesis* **.**

qed *simp*

lemma *mweak-conv4*:
assumes $\bigwedge A$. *closedin mtopology A* $\Longrightarrow$ *Limsup F* (λn. *measure* (*Ni n*) *A*) $\leq$ *measure N A*
and $\bigwedge U$. *openin mtopology U* $\Longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*)
and [*measurable*]: *A* $\in$ *sets* (*borel-of mtopology*)
and *measure N* (*mtopology frontier-of A*) = *0*
shows ((λn. *measure* (*Ni n*) *A*) $\longrightarrow$ *measure N A*) *F*
proof(*cases F* = $\bot$)
assume *F*: *F* $\neq \bot$
have [*measurable*]: *A* $\in$ *sets N mtopology closure-of A* $\in$ *sets N mtopology interior-of A* $\in$ *sets N*
mtopology frontier-of A $\in$ *sets N*
and *A*: $\forall_F$ *i in F*. *A* $\in$ *sets* (*Ni i*) $\forall_F$ *i in F*. *mtopology closure-of A* $\in$ *sets* (*Ni i*)
$\forall_F$ *i in F*. *mtopology interior-of A* $\in$ *sets* (*Ni i*) $\forall_F$ *i in F*. *mtopology frontier-of A* $\in$ *sets* (*Ni i*)
by(*auto simp*: *sets-N borel-of-open borel-of-closed closedin-frontier-of eventually-mono*[*OF sets-Ni*])
have *Limsup F* (λn. *measure* (*Ni n*) *A*) $\leq$ *Limsup F* (λn. *measure* (*Ni n*) (*mtopology closure-of A*))
using *sets.sets-into-space*[*OF assms*(*3*)]
by(*fastforce intro*!: *Limsup-mono finite-measure.finite-measure-mono*[*OF - closure-of-subset*]
eventually-mono[*OF eventually-conj*[*OF finite-measure-Ni A*(*2*)]] *simp*: *space-borel-of*)
also have ... $\leq$ *measure N* (*mtopology closure-of A*)
by(*auto intro*!: *assms*(*1*))
also have ... $\leq$ *measure N* (*A* $\cup$ (*mtopology frontier-of A*))
using *closure-of-subset*[*of A mtopology*] *sets.sets-into-space*[*OF assms*(*3*)] *interior-of-subset*[*of mtopology A*]
by(*auto simp*: *space-borel-of interior-of-union-frontier-of*[*symmetric*]
simp del: *interior-of-union-frontier-of intro*!: *N.finite-measure-mono*)
also have ... $\leq$ *measure N A* + *measure N* (*mtopology frontier-of A*)
by(*simp add*: *N.finite-measure-subadditive*)
also have ... = *measure N A* **by**(*simp add*: *assms*)
finally have *1*: *Limsup F* (λn. *measure* (*Ni n*) *A*) $\leq$ *measure N A* **.**
have *ereal* (*measure N A*) = *measure N A* $-$ *measure N* (*mtopology frontier-of A*)
by(*simp add*: *assms*)
also have ... $\leq$ *measure N* (*A* $-$ *mtopology frontier-of A*)
by(*auto simp*: *N.finite-measure-Diff'* *intro*!: *N.finite-measure-mono*)
also have ... $\leq$ *measure N* (*mtopology interior-of A*)
using *closure-of-subset*[*OF sets.sets-into-space*[*OF assms*(*3*),*simplified space-borel-of*]]
by(*auto intro*!: *N.finite-measure-mono simp*: *frontier-of-def*)
also have ... $\leq$ *Liminf F* (λn. *measure* (*Ni n*) (*mtopology interior-of A*))
by(*auto intro*!: *assms*)

also have ... $\leq$ *Liminf F* ($\lambda n.$ *measure* (*Ni n*) *A*)
by(*fastforce intro*!: *Liminf-mono finite-measure.finite-measure-mono interior-of-subset eventually-mono*[*OF eventually-conj*[*OF finite-measure-Ni A*(*1*)]]])
finally have *2*: *measure N A* $\leq$ *Liminf F* ($\lambda n.$ *measure* (*Ni n*) *A*) **.**
have *Liminf F* ($\lambda n.$ *measure* (*Ni n*) *A*) = *measure N A* $\wedge$ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) = *measure N A*
using *1 2 order.trans*[*OF 2 Liminf-le-Limsup*[*OF F*]] *order.trans*[*OF Liminf-le-Limsup*[*OF F*] *1*] *antisym*
by *blast*
thus *?thesis*
by (*metis F lim-ereal tendsto-Limsup*)
qed *simp*

lemma *mweak-conv5*:
assumes $\bigwedge A.$ $A \in$ *sets* (*borel-of mtopology*) $\Longrightarrow$ *measure N* (*mtopology frontier-of A*) = *0*
$\Longrightarrow$ (($\lambda n.$ *measure* (*Ni n*) *A*) $\longlongrightarrow$ *measure N A*) *F*
shows *mweak-conv Ni N F*
proof(*cases F* = $\bot$)
assume *F*: *F* $\neq \bot$
show *?thesis*
unfolding *mweak-conv-def*
proof *safe*
fix *f B*
assume *h*:*continuous-map mtopology euclideanreal f* $\forall x \in M.$ $|f\ x| \leq B$
have (($\lambda n.$ *measure* (*Ni n*) *M*) $\longlongrightarrow$ *measure N M*) *F*
using *frontier-of-topspace*[*of mtopology*] **by**(*auto intro*!: *assms borel-of-open*)
then obtain *K* **where** *K*: $\bigwedge A.$ $\forall_F$ *x in F*. *measure* (*Ni x*) *A* $\leq$ *K* $\bigwedge A.$ *measure N A* $\leq$ *K*
using *measure-converge-bounded'* **by** *metis*
from *continuous-map-measurable*[*OF h*(*1*)]
have *f*[*measurable*]:*f* $\in$ *borel-measurable N* $\forall_F$ *i in F*. *f* $\in$ *borel-measurable* (*Ni i*)
by(*auto cong*: *measurable-cong-sets simp*: *sets-N borel-of-euclidean intro*!: *eventually-mono*[*OF sets-Ni*])
have *f-int*[*simp*]: *integrable N f* $\forall_F$ *i in F*. *integrable* (*Ni i*) *f*
using *h* **by**(*auto intro*!: *N.integrable-const-bound*[**where** *B*=*B*] *finite-measure.integrable-const-bound*[**wher** *B*=*B*]
eventually-mono[*OF eventually-conj*[*OF eventually-conj*[*OF space-Ni f*(*2*)] *finite-measure-Ni*]] *simp*: *space-N*)
show (($\lambda n.$ $\int x.$ *f x* ∂*Ni n*) $\longlongrightarrow$ ($\int x.$ *f x* ∂N)) *F*
proof(*cases B* > *0*)
case *False*
with *h*(*2*) **have** *1*:$\bigwedge x.$ $x \in$ *space N* $\Longrightarrow$ *f x* = *0* $\forall_F$ *i in F*. $\forall x.$ $x \in$ *space* (*Ni i*) $\longrightarrow$ *f x* = *0*
by (*fastforce simp*: *space-N intro*!: *eventually-mono*[*OF space-Ni*])+
thus *?thesis*
by(*auto cong*: *Bochner-Integration.integral-cong*
intro!: *tendsto-cong*[**where** *g*=$\lambda x.$ *0* **and** *f*=($\lambda n.$ $\int x.$ *f x* ∂*Ni n*), *THEN*

```
iffD2] eventually-mono[OF 1(2)])
  next
    case B[arith]:True
    show ?thesis
    proof(cases K > 0)
      case False
      then have 1:measure N A = 0 ∀F x in F. measure (Ni x) M = 0 for A
        using K(2)[of A] measure-nonneg[of - A] measure-le-0-iff
        by(fastforce intro!: eventuallyI[THEN eventually-mp[OF - K(1)[of M]]])+
      hence N = null-measure (borel-of mtopology)
        by(auto intro!: measure-eqI simp: sets-N N.emeasure-eq-measure)
      moreover have ∀F x in F. Ni x = null-measure (borel-of mtopology)
        using order.trans[where c=0,OF finite-measure.bounded-measure]
          by(intro eventually-mono[OF eventually-conj[OF eventually-conj[OF space-Ni eventually-conj[OF finite-measure-Ni sets-Ni]] 1(2)]] measure-eqI)
          (auto simp: finite-measure.emeasure-eq-measure measure-le-0-iff)
      ultimately show ?thesis
        by (simp add: eventually-mono tendsto-eventually)
    next
      case [arith]:True
      show ?thesis
        unfolding tendsto-iff LIMSEQ-def dist-real-def
      proof safe
        fix r :: real
        assume r[arith]: r > 0
        define ν where ν ≡ distr N borel f
        have sets-nu[measurable-cong, simp]: sets ν = sets borel
          by(simp add: ν-def)
        interpret ν: finite-measure ν
          by(auto simp: ν-def N.finite-measure-distr)
        have (1 / 6) ∗ (r / K) ∗ (1 / B) > 0
          by auto
        from nat-approx-posE[OF this]
        obtain N′ where N′: 1 / (Suc N′) < (1 / 6) ∗ (r / K) ∗ (1 / B)
          by auto
        from mult-strict-right-mono[OF this B] have N′′:B / (Suc N′) < (1 / 6) ∗ (r / K)
          by auto
        have ∃ tn ∈ {B / Suc N′ ∗ (real n − 1) − B<..<B / Suc N′ ∗ real n − B}. measure ν {tn} = 0 for n
        proof(rule ccontr)
          assume ¬ (∃ tn ∈ {B / Suc N′ ∗ (real n − 1) − B<..<B / Suc N′ ∗ real n − B}. measure ν {tn} = 0)
          then have {B / Suc N′ ∗ (real n − 1) − B<..<B / Suc N′ ∗ real n − B} ⊆ {x. measure ν {x} ≠ 0}
            by auto
          moreover have uncountable {B / Suc N′ ∗ (real n − 1) − B<..<B / Suc N′ ∗ real n − B}
            unfolding uncountable-open-interval right-diff-distrib by auto
```

ultimately show *False*
using ν*.countable-support* **by**(*meson countable-subset*)
qed
then obtain *tn* **where** *tn*: $\bigwedge n.\ B\ /\ Suc\ N' * (real\ n - 1) - B < tn\ n$
$\bigwedge n.\ tn\ n < B\ /\ Suc\ N' * real\ n - B$
$\bigwedge n.\ measure\ \nu\ \{tn\ n\} = 0$
by (*metis greaterThanLessThan-iff*)
have *t0*: $tn\ 0 < - B$
using *tn*(*2*)[*of 0*] **by** *simp*
have *tN*: $B < tn\ (Suc\ (2 * (Suc\ N')))$
proof −
have $B * (2 + 2 * real\ N')\ /\ (1 + real\ N') = 2 * B$
by(*auto simp*: *divide-eq-eq*)
with *tn*(*1*)[*of* $Suc\ (2 * (Suc\ N'))$] **show** *?thesis*
by *simp*
qed
define *Aj* **where** $Aj \equiv (\lambda j.\ f -` \{tn\ j..<tn\ (Suc\ j)\} \cap M)$
have *sets-Aj*[*measurable*]: $\bigwedge j.\ Aj\ j \in sets\ N$ $\forall_F\ i\ in\ F.\ \forall j.\ Aj\ j \in sets\ (Ni\ i)$
using *measurable-sets*[*OF f*(*1*)]
by(*auto simp*: *Aj-def space-N intro*!: *eventually-mono*[*OF eventually-conj*[*OF space-Ni f*(*2*)]])
have *m-f*: $measure\ N\ (mtopology\ frontier\text{-}of\ (Aj\ j)) = 0$ **for** *j*
proof −
have $measure\ N\ (mtopology\ frontier\text{-}of\ (Aj\ j)) = measure\ N\ (mtopology\ closure\text{-}of\ (Aj\ j) - mtopology\ interior\text{-}of\ (Aj\ j))$
by(*simp add*: *frontier-of-def*)
also have ... $\le measure\ \nu\ \{tn\ j,\ tn\ (Suc\ j)\}$
proof −
have [*simp*]: $\{x \in M.\ tn\ j \le f\ x \wedge f\ x \le tn\ (Suc\ j)\} = f -` \{tn\ j..tn\ (Suc\ j)\} \cap M$
$\{x \in M.\ tn\ j < f\ x \wedge f\ x < tn\ (Suc\ j)\} = f -` \{tn\ j<..<tn\ (Suc\ j)\} \cap M$
by *auto*
have $mtopology\ closure\text{-}of\ (Aj\ j) \subseteq f -` \{tn\ j..tn\ (Suc\ j)\} \cap M$
by(*rule closure-of-minimal*,*insert closedin-continuous-map-preimage*[*OF h*(*1*),*of* $\{tn\ j..tn\ (Suc\ j)\}$])
(*auto simp*: *Aj-def*)
moreover have $f -` \{tn\ j<..<tn\ (Suc\ j)\} \cap M \subseteq mtopology\ interior\text{-}of\ (Aj\ j)$
by(*rule interior-of-maximal*,*insert openin-continuous-map-preimage*[*OF h*(*1*),*of* $\{tn\ j<..<tn\ (Suc\ j)\}$])
(*auto simp*: *Aj-def*)
ultimately have $mtopology\ closure\text{-}of\ (Aj\ j) - mtopology\ interior\text{-}of\ (Aj\ j) \subseteq f -` \{tn\ j,tn\ (Suc\ j)\} \cap M$
by(*fastforce dest*: *contra-subsetD*)
with *closedin-subset*[*OF closedin-closure-of*,*of mtopology Aj j*] **show** *?thesis*
by(*auto simp*: ν*-def measure-distr intro*!: *N.finite-measure-mono*)

(*auto simp*: *space-N*)
qed
also have ... ≤ *measure* ν {*tn j*} + *measure* ν {*tn* (*Suc j*)}
using ν.*finite-measure-subadditive*[*of* {*tn* (*Suc j*)} {*tn j*}] **by** *auto*
also have ... = *0*
by(*simp add*: *tn*)
finally show *?thesis*
by (*simp add*: *measure-le-0-iff*)
qed
hence *conv*:((λ*n*. *measure* (*Ni n*) (*Aj j*)) ⟶ *measure N* (*Aj j*)) *F* **for** *j*
by(*auto intro*!: *assms simp*: *sets-N*[*symmetric*] *sets-Ni*)
have *fil1*:∀$_F$ *n in F*. |*tn j*| ∗ |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)| < *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc N'*)))) **for** *j*
proof(*cases* |*tn j*| = *0*)
case *pos*:*False*
then have *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc N'*)))) ∗ (*1* / |*tn j*|) > *0*
by *auto*
with *conv*[*of j*]
have *1*:∀$_F$ *n in F*. |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)|
< *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc N'*)))) ∗ (*1* / |*tn j*|)
unfolding *tendsto-iff dist-real-def* **by** *metis*
have ∀$_F$ *n in F*. |*tn j*| ∗ |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)| < *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc N'*))))
proof(*rule eventuallyI*[*THEN eventually-mp*[*OF* - *1*]])
show |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)| < *r* / *real* (*3* ∗ *Suc* (*Suc* (*2* ∗ *Suc N'*))) ∗ (*1* / |*tn j*|)
⟶ |*tn j*| ∗ |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)| < *r* / *real* (*3* ∗ *Suc* (*Suc* (*2* ∗ *Suc N'*))) **for** *n*
using *mult-less-cancel-right-pos*[*of* |*tn j*| |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)|
r / *real* (*3* ∗ *Suc* (*Suc* (*2* ∗ *Suc N'*))) ∗ (*1* / |*tn j*|)] *pos* **by**(*simp add*: *mult.commute*)
qed
thus *?thesis* **by** *auto*
qed *auto*
hence *fil1*:∀$_F$ *n in F*. ∀*j*∈{..*Suc* (*2* ∗ *Suc N'*)}. |*tn j*| ∗ |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)|
< *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc N'*))))
by(*auto intro*!: *eventually-ball-finite*)
have *tn-strictmono*: *strict-mono tn*
unfolding *strict-mono-Suc-iff*
proof *safe*
fix *n*
show *tn n* < *tn* (*Suc n*)
using *tn*(*1*)[*of Suc n*] *tn*(*2*)[*of n*] **by** *auto*
qed
from *strict-mono-less*[*OF this*] **have** *Aj-disj*: *disjoint-family Aj*
by(*auto simp*: *disjoint-family-on-def Aj-def*) (*metis linorder-not-le not-less-eq order-less-le order-less-trans*)

have *Aj-un*: $M = (\bigcup i{\in}\{..Suc\ (2 * Suc\ N')\}.\ Aj\ i)$
proof
show $M \subseteq \bigcup\ (Aj\ `\ \{..Suc\ (2 * Suc\ N')\})$
proof
fix *x*
assume $x{:}x \in M$
with *h*(*2*) *tN t0* **have** $h'{:}tn\ 0 < f\ x\ f\ x < tn\ (Suc\ (2 * Suc\ N'))$
by *fastforce+*
define *n* **where** $n \equiv LEAST\ n.\ f\ x < tn\ (Suc\ n)$
have $f\ x < tn\ (Suc\ n)$
unfolding *n-def* **by**(*rule LeastI-ex*) (*use* h' **in** *auto*)
moreover have $tn\ n \le f\ x$
by (*metis Least-le Suc-n-not-le-n* $h'(1)$ *less-eq-real-def linorder-not-less n-def not0-implies-Suc*)
moreover have $n \le 2 * Suc\ N'$
unfolding *n-def* **by**(*rule Least-le*) (*use* h' **in** *auto*)
ultimately show $x \in \bigcup\ (Aj\ `\ \{..Suc\ (2 * Suc\ N')\})$
by(*auto simp*: *Aj-def x*)
qed
qed(*auto simp*: *Aj-def*)
define *h* **where** $h \equiv (\lambda x.\ \sum i{\le}Suc\ (2 * (Suc\ N')).\ tn\ i * indicat\text{-}real\ (Aj\ i)\ x)$
have *h*[*measurable*]: $h \in borel\text{-}measurable\ N\ \forall_F\ i\ in\ F.\ h \in borel\text{-}measurable\ (Ni\ i)$
by(*auto simp*: *h-def simp del*: *sum.atMost-Suc sum-mult-indicator intro*!: *borel-measurable-sum eventually-mono*[*OF sets-Aj*(*2*)])
have *h-f*: $h\ x \le f\ x$ **if** $x \in M$ **for** *x*
proof −
from *that disjoint-family-onD*[*OF Aj-disj*]
obtain *n* **where** *n*: $x \in Aj\ n\ n \le Suc\ (2 * Suc\ N')\ \bigwedge m.\ m \ne n \Longrightarrow x \notin Aj\ m$
by(*auto simp*: *Aj-un*)
have $h\ x = (\sum i{\le}Suc\ (2 * (Suc\ N')).\ if\ i = n\ then\ tn\ i\ else\ 0)$
unfolding *h-def* **by**(*rule Finite-Cartesian-Product.sum-cong-aux*) (*use n* **in** *auto*)
also have ... $= tn\ n$
using *n* **by** *auto*
also have ... $\le f\ x$
using *n*(*1*) **by**(*auto simp*: *Aj-def*)
finally show *?thesis* **.**
qed
have *f-h*: $f\ x < h\ x + (1\ /\ 3) * (r\ /\ enn2real\ K)$ **if** $x \in M$ **for** *x*
proof −
from *that disjoint-family-onD*[*OF Aj-disj*]
obtain *n* **where** *n*: $x \in Aj\ n\ n \le Suc\ (2 * Suc\ N')\ \bigwedge m.\ m \ne n \Longrightarrow x \notin Aj\ m$
by(*auto simp*: *Aj-un*)
have $h\ x = (\sum i{\le}Suc\ (2 * (Suc\ N')).\ if\ i = n\ then\ tn\ i\ else\ 0)$
unfolding *h-def* **by**(*rule Finite-Cartesian-Product.sum-cong-aux*) (*use*

n **in** *auto*)
also have ... = *tn n*
using *n* **by** *auto*
finally have *hx*: *h x* = *tn n* **.**
have *f x* < *tn* (*Suc n*)
using *n* **by**(*auto simp*: *Aj-def*)
hence *f x* − *tn n* < *tn* (*Suc n*) − *tn n* **by** *auto*
also have ... < *B* / *real* (*Suc N′*) ∗ *real* (*Suc n*) − (*B* / *real* (*Suc N′*) ∗ (*real n* − *1*))
using *tn*(*1*)[*of n*] *tn*(*2*)[*of Suc n*] **by** *auto*
also have ... = *2* ∗ *B* / *real* (*Suc N′*)
by(*auto simp*: *diff-divide-distrib*[*symmetric*]) (*simp add*: *ring-distribs*(*1*) *right-diff-distrib*)
also have ... < (*1* / *3*) ∗ (*r* / *enn2real K*)
using *N′′* **by** *auto*
finally show *?thesis*
using *hx* **by** *simp*
qed
with *h-f* **have** *fh*:⋀*x*. *x* ∈ *M* ⟹ |*f x* − *h x*| < (*1* / *3*) ∗ (*r* / *enn2real K*)
by *fastforce*
have *h-bounded*: |*h x*| ≤ (∑ *i*≤*Suc* (*2* ∗ (*Suc N′*)). |*tn i*|) **for** *x*
unfolding *h-def* **by**(*rule order.trans*[*OF sum-abs*[*of* λ*i*. *tn i* ∗ *indicat-real* (*Aj i*) *x*
{..*Suc* (*2* ∗ (*Suc N′*))}] *sum-mono*]) (*auto simp*: *indicator-def*)
hence *h-int*[*simp*]: *integrable N h* ∀$_F$ *i in F*. *integrable* (*Ni i*) *h*
by(*auto intro*!: *N*.*integrable-const-bound*[**where** *B*=∑ *i*≤*Suc* (*2* ∗ (*Suc N′*)). |*tn i*|]
finite-measure.*integrable-const-bound*[**where** *B*=∑ *i*≤*Suc* (*2* ∗ (*Suc N′*)). |*tn i*|]
eventually-mono[*OF eventually-conj*[*OF finite-measure-Ni h*(*2*)]]])
show ∀$_F$ *n in F*. |(∫ *x*. *f x* ∂*Ni n*) − (∫ *x*. *f x* ∂*N*)| < *r*
proof(*safe intro*!: *eventually-mono*[*OF eventually-conj*[*OF K*(*1*)[*of M*]
eventually-conj[*OF eventually-conj*[*OF fil1 h-int*(*2*)]
eventually-conj[*OF f-int*(*2*)
eventually-conj[*OF eventually-conj*[*OF finite-measure-Ni space-Ni*]
sets-Aj(*2*)]]]]]])
fix *n*
assume *n*:∀ *j*∈{..*Suc* (*2* ∗ *Suc N′*)}.
|*tn j*| ∗ |*measure* (*Ni n*) (*Aj j*) − *measure N* (*Aj j*)| < *r* / *real* (*3* ∗ *Suc* (*Suc* (*2* ∗ *Suc N′*)))
measure (*Ni n*) (*space* (*Ni n*)) ≤ *K*
and *h-intn*[*simp*]:*integrable* (*Ni n*) *h* **and** *f-intn*[*simp*]:*integrable* (*Ni n*) *f*
and *sets-Aj2*[*measurable*]:∀ *j*. *Aj j* ∈ *sets* (*Ni n*)
and *space-Ni*:*M* = *space* (*Ni n*)
and *finite-measure* (*Ni n*)
interpret *Ni*: *finite-measure* (*Ni n*) **by** *fact*

have |(∫ x. f x ∂Ni n) − (∫ x. f x ∂N)|
= |(∫ x. f x − h x ∂Ni n) + ((∫ x. h x ∂Ni n) − (∫ x. h x ∂N)) − (∫ x. f x − h x ∂N)|
by(*simp add*: *Bochner-Integration.integral-diff*[*OF f-int*(*1*) *h-int*(*1*)] *Bochner-Integration.integral-diff*[*OF f-intn h-intn*])
also have ... ≤ |∫ x. f x − h x ∂Ni n| + |(∫ x. h x ∂Ni n) − (∫ x. h x ∂N)| + |∫ x. f x − h x ∂N|
by *linarith*
also have ... ≤ (∫ x. |f x − h x| ∂Ni n) + |(∫ x. h x ∂Ni n) − (∫ x. h x ∂N)| + (∫ x. |f x − h x| ∂N)
using *integral-abs-bound* **by** (*simp add*: *add-mono del*: *f-int f-intn*)
also have ... ≤ r / 3 + |(∫ x. h x ∂Ni n) − (∫ x. h x ∂N)| + r / 3
proof −
have (∫ x. |f x − h x| ∂Ni n) ≤ (∫ x. (1 / 3) ∗ (r / enn2real K) ∂Ni n)
by(*rule integral-mono*) (*insert fh*, *auto simp*: *space-Ni order.strict-implies-order*)
also have ... = *measure* (*Ni n*) (*space* (*Ni n*)) / K ∗ (r / 3)
by *auto*
also have ... ≤ r / 3
by(*rule mult-left-le-one-le*) (*use n space-Ni* **in** *auto*)
finally have *1*:(∫ x. |f x − h x| ∂Ni n) ≤ r / 3 **.**
have (∫ x. |f x − h x| ∂N) ≤ (∫ x. (1 / 3) ∗ (r / K) ∂N)
by(*rule integral-mono*) (*insert fh*, *auto simp*: *space-N order.strict-implies-order*)
also have ... = *measure N* (*space N*) / *enn2real K* ∗ (r / 3)
by *auto*
also have ... ≤ r / 3
by(*rule mult-left-le-one-le*) (*use K space-N* **in** *auto*)
finally show *?thesis*
using *1* **by** *auto*
qed
also have ... < r
proof −
have |(∫ x. h x ∂Ni n) − (∫ x. h x ∂N)|
= |(∫ x. (∑ i≤Suc (2 ∗ (Suc N′)). tn i ∗ indicat-real (Aj i) x) ∂Ni n)
− (∫ x. (∑ i≤Suc (2 ∗ (Suc N′)). tn i ∗ indicat-real (Aj i) x) ∂N)|
by(*simp add*: *h-def*)
also have ... = |(∑ i≤Suc (2 ∗ (Suc N′)). (∫ x. tn i ∗ indicat-real (Aj i) x ∂Ni n))
− (∑ i≤Suc (2 ∗ (Suc N′)). (∫ x. tn i ∗ indicat-real (Aj i) x ∂N))|
proof −
have *1*: (∫ x. (∑ i≤Suc (2 ∗ (Suc N′)). tn i ∗ indicat-real (Aj i) x) ∂Ni n)
= (∑ i≤Suc (2 ∗ (Suc N′)). (∫ x. tn i ∗ indicat-real (Aj i) x ∂Ni n))
by(*rule Bochner-Integration.integral-sum*) (*use integrable-real-mult-indicator sets-Aj2* **in** *blast*)
have *2*: (∫ x. (∑ i≤Suc (2 ∗ (Suc N′)). tn i ∗ indicat-real (Aj i) x)

∂N)
= ($\sum$ $i \le Suc$ (2 ∗ (Suc N')). ($\int$ x. tn i ∗ *indicat-real* (Aj i) x ∂N))
by(*rule Bochner-Integration.integral-sum*) (*use integrable-real-mult-indicator sets-Aj*(*1*) **in** *blast*)
show *?thesis*
by(*simp only*: *1 2*)
qed
also have ... = |($\sum$ $i \le Suc$ (2 ∗ (Suc N')). *tn i* ∗ *measure* (*Ni n*) (*Aj i*))
− ($\sum$ $i \le Suc$ (2 ∗ (Suc N')). *tn i* ∗ *measure N* (*Aj i*))|
by *simp*
also have ... = |$\sum$ $i \le Suc$ (2 ∗ (Suc N')). *tn i* ∗ (*measure* (*Ni n*) (*Aj i*) − *measure N* (*Aj i*))|
by(*auto simp*: *sum-subtractf right-diff-distrib*)
also have ... ≤ ($\sum$ $i \le Suc$ (2 ∗ (Suc N')). |*tn i* ∗ (*measure* (*Ni n*) (*Aj i*) − *measure N* (*Aj i*))|)
by(*rule sum-abs*)
also have ... ≤ ($\sum$ $i \le Suc$ (2 ∗ (Suc N')). |*tn i*| ∗ |(*measure* (*Ni n*) (*Aj i*) − *measure N* (*Aj i*))|)
by(*simp add*: *abs-mult*)
also have ... < ($\sum$ $i \le Suc$ (2 ∗ (Suc N')). *r* / (*3* ∗ (*Suc* (*Suc* (*2* ∗ *Suc* N')))))
by(*rule sum-strict-mono*) (*use n* **in** *auto*)
also have ... = *real* (*Suc* (*Suc* (*2* ∗ *Suc* N'))) ∗ (*1* / (*Suc* (*Suc* (*2* ∗ *Suc* N'))) ∗ (*r* / *3*))
by *auto*
also have ... = *r* / *3*
unfolding *mult.assoc*[*symmetric*] **by** *simp*
finally show *?thesis* **by** *auto*
qed
finally show |($\int$ x. f x ∂Ni n) − ($\int$ x. f x ∂N)| < r **.**
qed
qed
qed
qed
qed(*auto simp*: *sets-N finite-measure-N intro*!: *eventually-mono*[*OF eventually-Ni*])
qed (*simp add*: *mweak-conv-def sets-Ni sets-N finite-measure-N*)

lemma *mweak-conv-eq*: *mweak-conv Ni N F*
⟷ (∀ f::'a ⇒ *real*. *continuous-map mtopology euclidean f* ⟶ (∃ B. ∀ $x \in M$. |f x| ≤ B)
⟶ ((λn. $\int$ x. f x ∂Ni n) ⟶ ($\int$ x. f x ∂N)) F)
by(*auto simp*: *sets-N mweak-conv-def finite-measure-N*
intro!: *eventually-mono*[*OF eventually-conj*[*OF finite-measure-Ni sets-Ni*]])

lemma *mweak-conv-eq1*: *mweak-conv Ni N F*
⟷ (∀ f::'a ⇒ *real*. *uniformly-continuous-map Self euclidean-metric f* ⟶ (∃ B. ∀ $x \in M$. |f x| ≤ B)

$\longrightarrow ((\lambda n.\ \int x.\ f\ x\ \partial Ni\ n) \longrightarrow (\int x.\ f\ x\ \partial N))\ F)$
proof
assume *h*:$\forall f$::*'a* ⇒ *real. uniformly-continuous-map Self euclidean-metric f* ⟶ ($\exists B.\ \forall x \in M.\ |f\ x| \le B$)
⟶ (($\lambda n.\ \int x.\ f\ x\ \partial Ni\ n$) ⟶ ($\int x.\ f\ x\ \partial N$)) *F*
have *1*:(($\lambda n.$ *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
proof −
have *1*:(($\lambda n.$ *measure* (*Ni n*) (*space* (*Ni n*))) ⟶ *measure N M*) *F*
using *h*[*rule-format*,*OF uniformly-continuous-map-const*[*THEN iffD2*,*of - 1*]]
by(*auto simp*: *space-N*)
show *?thesis*
by(*auto intro*!: *tendsto-cong*[*THEN iffD1*,*OF - 1*] *eventually-mono*[*OF space-Ni*])
qed
have ⋀*A. closedin mtopology A* ⟹ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) ≤ *measure N A*
and ⋀*U. openin mtopology U* ⟹ *measure N U* ≤ *Liminf F* ($\lambda n.$ *measure* (*Ni n*) *U*)
using *mweak-conv2*[*OF h*[*rule-format*]] *mweak-conv3*[*OF - 1*] **by** *auto*
hence ⋀*A. A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure N* (*mtopology frontier-of A*) = *0*
⟹ (($\lambda n.$ *measure* (*Ni n*) *A*) ⟶ *measure N A*) *F*
using *mweak-conv4* **by** *auto*
with *mweak-conv5* **show** *mweak-conv Ni N F* **by** *auto*
qed(*use mweak-conv1* **in** *auto*)

lemma *mweak-conv-eq2*: *mweak-conv Ni N F*
⟷ (($\lambda n.$ *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F* ∧ ($\forall A.$ *closedin mtopology A*
⟶ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) ≤ *measure N A*)
proof *safe*
assume *mweak-conv Ni N F*
note *h* = *this*[*simplified mweak-conv-eq1*]
show *1*:(($\lambda n.$ *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
proof −
have *1*:(($\lambda n.$ *measure* (*Ni n*) (*space* (*Ni n*))) ⟶ *measure N M*) *F*
using *h*[*rule-format*,*OF uniformly-continuous-map-const*[*THEN iffD2*,*of - 1*]]
by(*auto simp*: *space-N*)
show *?thesis*
by(*auto intro*!: *tendsto-cong*[*THEN iffD1*,*OF - 1*] *eventually-mono*[*OF space-Ni*])
qed
show ⋀*A. closedin mtopology A* ⟹ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) ≤ *measure N A*
using *mweak-conv2*[*OF h*[*rule-format*]] **by** *auto*
next
assume *h*: (($\lambda n.$ *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
$\forall A.$ *closedin mtopology A* ⟶ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) ≤ *measure N A*
then have ⋀*A. closedin mtopology A* ⟹ *Limsup F* ($\lambda n.$ *measure* (*Ni n*) *A*) ≤ *measure N A*

and $\bigwedge U$. *openin mtopology U* $\Longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*)
using *mweak-conv3* **by** *auto*
hence $\bigwedge A$. $A \in$ *sets* (*borel-of mtopology*) $\Longrightarrow$ *measure N* (*mtopology frontier-of A*) = *0*
$\Longrightarrow$ ((λn. *measure* (*Ni n*) *A*) $\longrightarrow$ *measure N A*) *F*
using *mweak-conv4* **by** *auto*
with *mweak-conv5* **show** *mweak-conv Ni N F* **by** *auto*
qed

lemma *mweak-conv-eq3*: *mweak-conv Ni N F*
$\longleftrightarrow$ ((λn. *measure* (*Ni n*) *M*) $\longrightarrow$ *measure N M*) *F* $\wedge$
($\forall$ *U*. *openin mtopology U* $\longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*))
proof *safe*
assume *mweak-conv Ni N F*
note *h* = *this*[*simplified mweak-conv-eq1*]
show *1*:((λn. *measure* (*Ni n*) *M*) $\longrightarrow$ *measure N M*) *F*
proof –
have *1*:((λn. *measure* (*Ni n*) (*space* (*Ni n*))) $\longrightarrow$ *measure N M*) *F*
using *h*[*rule-format*,*OF uniformly-continuous-map-const*[*THEN iffD2*,*of - 1*]]
by(*auto simp*: *space-N*)
show *?thesis*
by(*auto intro*!: *tendsto-cong*[*THEN iffD1*,*OF - 1*] *eventually-mono*[*OF space-Ni*])
qed
show $\bigwedge U$. *openin mtopology U* $\Longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*)
using *mweak-conv2*[*OF h*[*rule-format*]] *mweak-conv3*[*OF - 1*] **by** *auto*
next
assume *h*: ((λn. *measure* (*Ni n*) *M*) $\longrightarrow$ *measure N M*) *F*
$\forall$ *U*. *openin mtopology U* $\longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*)
then have $\bigwedge A$. *closedin mtopology A* $\Longrightarrow$ *Limsup F* (λn. *measure* (*Ni n*) *A*) $\leq$ *measure N A*
and $\bigwedge U$. *openin mtopology U* $\Longrightarrow$ *measure N U* $\leq$ *Liminf F* (λn. *measure* (*Ni n*) *U*)
using *mweak-conv3'* **by** *auto*
hence $\bigwedge A$. $A \in$ *sets* (*borel-of mtopology*) $\Longrightarrow$ *measure N* (*mtopology frontier-of A*) = *0*
$\Longrightarrow$ ((λn. *measure* (*Ni n*) *A*) $\longrightarrow$ *measure N A*) *F*
using *mweak-conv4* **by** *auto*
with *mweak-conv5* **show** *mweak-conv Ni N F* **by** *auto*
qed

lemma *mweak-conv-eq4*: *mweak-conv Ni N F*
$\longleftrightarrow$ ($\forall$ $A \in$ *sets* (*borel-of mtopology*). *measure N* (*mtopology frontier-of A*) = *0*
$\longrightarrow$ ((λn. *measure* (*Ni n*) *A*) $\longrightarrow$ *measure N A*) *F*)
proof *safe*
assume *mweak-conv Ni N F*

note *h* = *this*[*simplified mweak-conv-eq1*]
have *1*:((λ*n*. *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
proof −
have *1*:((λ*n*. *measure* (*Ni n*) (*space* (*Ni n*))) ⟶ *measure N M*) *F*
using *h*[*rule-format*,*OF uniformly-continuous-map-const*[*THEN iffD2*,*of* - *1*]]
by(*auto simp*: *space-N*)
show *?thesis*
by(*auto intro*!: *tendsto-cong*[*THEN iffD1*,*OF* - *1*] *eventually-mono*[*OF space-Ni*])
qed
have ⋀*A*. *closedin mtopology A* ⟹ *Limsup F* (λ*n*. *measure* (*Ni n*) *A*) ≤ *measure N A*
and ⋀*U*. *openin mtopology U* ⟹ *measure N U* ≤ *Liminf F* (λ*n*. *measure* (*Ni n*) *U*)
using *mweak-conv2*[*OF h*[*rule-format*]] *mweak-conv3*[*OF* - *1*] **by** *auto*
thus ⋀*A*. *A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure N* (*mtopology frontier-of A*) = *0*
⟹ ((λ*n*. *measure* (*Ni n*) *A*) ⟶ *measure N A*) *F*
using *mweak-conv4* **by** *auto*
qed(*use mweak-conv5* **in** *auto*)

corollary *mweak-conv-imp-limit-space*:
assumes *mweak-conv Ni N F*
shows ((λ*i*. *measure* (*Ni i*) *M*) ⟶ *measure N M*) *F*
using *assms* **by**(*simp add*: *mweak-conv-eq3*)

end

lemma
assumes *metrizable-space X*
and $\forall_F$ *i in F*. *sets* (*Ni i*) = *sets* (*borel-of X*) $\forall_F$ *i in F*. *finite-measure* (*Ni i*)
and *sets N* = *sets* (*borel-of X*) *finite-measure N*
shows *weak-conv-on-eq1*:
weak-conv-on Ni N F X
⟷ ((λ*n*. *measure* (*Ni n*) (*topspace X*)) ⟶ *measure N* (*topspace X*)) *F*
∧ (∀ *A*. *closedin X A* ⟶ *Limsup F* (λ*n*. *measure* (*Ni n*) *A*) ≤ *measure N A*) (**is** *?eq1*)
and *weak-conv-on-eq2*:
weak-conv-on Ni N F X
⟷ ((λ*n*. *measure* (*Ni n*) (*topspace X*)) ⟶ *measure N* (*topspace X*)) *F*
∧ (∀ *U*. *openin X U* ⟶ *measure N U* ≤ *Liminf F* (λ*n*. *measure* (*Ni n*) *U*)) (**is** *?eq2*)
and *weak-conv-on-eq3*:
weak-conv-on Ni N F X
⟷ (∀ *A* ∈ *sets* (*borel-of X*). *measure N* (*X frontier-of A*) = *0*
⟶ ((λ*n*. *measure* (*Ni n*) *A*) ⟶ *measure N A*) *F*) (**is** *?eq3*)
proof −
obtain *d* **where** *d*: *Metric-space* (*topspace X*) *d Metric-space.mtopology* (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms*(*1*) *metrizable-space-def*)

then interpret *mweak-conv-fin topspace X d Ni N*
by(*auto simp*: *mweak-conv-fin-def mweak-conv-fin-axioms-def assms*)
show *?eq1 ?eq2 ?eq3*
using *mweak-conv-eq2 mweak-conv-eq3 mweak-conv-eq4* **unfolding** *d*(*2*) **by** *blast+*
qed

end

4 The Lévy-Prokhorov Metric

theory *Levy-Prokhorov-Distance*
imports *Lemmas-Levy-Prokhorov General-Weak-Convergence*
begin

4.1 The Lévy-Prokhorov Metric

lemma *LPm-ne′*:
assumes *finite-measure M finite-measure N*
shows ∃ *e*>*0*. ∀ *A B C D*. *measure M A* ≤ *measure N* (*B A e*) + *e* ∧ *measure N C* ≤ *measure M* (*D C e*) + *e*
proof −
interpret *M*: *finite-measure M* **by** *fact*
interpret *N*: *finite-measure N* **by** *fact*
from *M*.*emeasure-real N*.*emeasure-real* **obtain** *m n* **where** *mn*[*arith*]:
m ≥ *0 n* ≥ *0 M* (*space M*) = *ennreal m N* (*space N*) = *ennreal n*
by *metis*
then have *MN*:⋀*A*. *measure M A* ≤ *m* ⋀*A*. *measure N A* ≤ *n*
using *M*.*bounded-measure N*.*bounded-measure measure-eq-emeasure-eq-ennreal* **by** *blast+*
show *?thesis*
proof(*safe intro*!: *exI*[**where** *x*=*m*+*n*+*1*])
fix *A B C D*
note [*arith*] = *MN*(*1*)[*of A*] *MN*(*1*)[*of D C* (*m* + *n* + *1*)] *MN*(*2*)[*of C*] *MN*(*2*)[*of B A* (*m* + *n* + *1*)]
show *measure M A* ≤ *measure N* (*B A* (*m*+*n*+*1*)) + (*m*+*n*+*1*) *measure N C* ≤ *measure M* (*D C* (*m*+*n*+*1*)) + (*m*+*n*+*1*)
by(*simp-all add*: *add*.*commute add-increasing2*)
qed *simp*
qed

locale *Levy-Prokhorov* = *Metric-space*
begin

definition 𝒫 ≡ {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *finite-measure N*}

lemma *inP-D*:
assumes *N* ∈ 𝒫
shows *finite-measure N sets N* = *sets* (*borel-of mtopology*) *space N* = *M*

using *assms* **by**(*auto simp*: $\mathcal{P}$*-def space-borel-of cong*: *sets-eq-imp-space-eq*)

declare *inP-D*(*2*)[*measurable-cong*]

lemma *inP-I*: *sets N* = *sets* (*borel-of mtopology*) $\Longrightarrow$ *finite-measure N* $\Longrightarrow$ *N* $\in$ $\mathcal{P}$
by(*auto simp*: $\mathcal{P}$*-def*)

lemma *inP-iff*: *N* $\in \mathcal{P} \longleftrightarrow$ *sets N* = *sets* (*borel-of mtopology*) $\wedge$ *finite-measure N*
by(*simp add*: $\mathcal{P}$*-def*)

lemma *M-empty-P*:
assumes *M* = {}
shows $\mathcal{P}$ = {} $\vee$ $\mathcal{P}$ = {*count-space* {}}
proof −
have $\bigwedge N$. *N* $\in \mathcal{P} \Longrightarrow$ *N* = *count-space* {}
by (*simp add*: *assms inP-D*(*3*) *space-empty*)
thus *?thesis*
by *blast*
qed

lemma *M-empty-P'*:
assumes *M* = {}
shows $\mathcal{P}$ = {} $\vee$ $\mathcal{P}$ = {*null-measure* (*borel-of mtopology*)}
by (*metis inP-D*(*2*) *singletonI space-count-space space-empty space-empty-iff space-null-measure M-empty-P*[*OF assms*])

lemma *inP-mweak-conv-fin-all*:
assumes $\bigwedge i$. *Ni i* $\in \mathcal{P}$ *N* $\in \mathcal{P}$
shows *mweak-conv-fin M d Ni N F*
using *assms inP-D* **by**(*auto simp*: *mweak-conv-fin-def Metric-space-axioms mweak-conv-fin-axioms-def*)

lemma *inP-mweak-conv-fin*:
assumes $\forall_F$ *i in F*. *Ni i* $\in \mathcal{P}$ *N* $\in \mathcal{P}$
shows *mweak-conv-fin M d Ni N F*
using *assms inP-D* **by**(*auto simp*: *mweak-conv-fin-def Metric-space-axioms mweak-conv-fin-axioms-def intro*!: *eventually-mono*[*OF assms*(*1*)])

definition *LPm* :: *'a measure* $\Rightarrow$ *'a measure* $\Rightarrow$ *real* **where**
LPm N L $\equiv$
if N $\in \mathcal{P} \wedge$ *L* $\in \mathcal{P}$ *then*
($\bigsqcap$ {*e*. *e* > *0* $\wedge$ ($\forall$ *A*$\in$*sets* (*borel-of mtopology*).
measure N A $\leq$ *measure L* ($\bigcup$ *a*$\in$*A*. *mball a e*) + *e* $\wedge$
measure L A $\leq$ *measure N* ($\bigcup$ *a*$\in$*A*. *mball a e*) + *e*)})
else 0

lemma *bdd-below-Levy-Prokhorov*:
bdd-below {*e*. *e* > *0* $\wedge$ ($\forall$ *A*$\in$*sets* (*borel-of mtopology*).
measure N A $\leq$ *measure L* ($\bigcup$ *a*$\in$*A*. *mball a e*) + *e* $\wedge$

$measure\ L\ A \le measure\ N\ (\bigcup a{\in}A.\ mball\ a\ e) + e)\}$
by(*auto intro*!: *bdd-belowI*[**where** $m{=}0$])

lemma *LPm-ne*:
assumes $N \in \mathcal{P}$ $L \in \mathcal{P}$
shows $\{e.\ e > 0 \wedge (\forall A{\in}sets\ (borel\text{-}of\ mtopology).$
$measure\ N\ A \le measure\ L\ (\bigcup a{\in}A.\ mball\ a\ e) + e \wedge$
$measure\ L\ A \le measure\ N\ (\bigcup a{\in}A.\ mball\ a\ e) + e)\}$
$\neq \{\}$
proof −
from *LPm-ne′*[*OF inP-D*(*1*)[*OF assms*(*1*)] *inP-D*(*1*)[*OF assms*(*2*)]]
show *?thesis* **by** *fastforce*
qed

lemma *LPm-imp-le*:
assumes $e > 0$
and $\bigwedge B.\ B \in sets\ (borel\text{-}of\ mtopology) \Longrightarrow measure\ L\ B \le measure\ N\ (\bigcup a{\in}B.\ mball\ a\ e) + e$
and $\bigwedge B.\ B \in sets\ (borel\text{-}of\ mtopology) \Longrightarrow measure\ N\ B \le measure\ L\ (\bigcup a{\in}B.\ mball\ a\ e) + e$
shows $LPm\ L\ N \le e$
proof −
consider $L \notin \mathcal{P} \mid N \notin \mathcal{P} \mid L \in \mathcal{P}\ N \in \mathcal{P}$ **by** *auto*
then show *?thesis*
proof *cases*
case *3*
show *?thesis*
by(*auto simp add*: *LPm-def 3 intro*!: *cINF-lower*[**where** *f=id*,*simplified*] *assms bdd-belowI*[**where** $m{=}0$])
qed(*insert assms*,*simp-all add*: *LPm-def*)
qed

lemma *LPm-le-max-measure*: $LPm\ L\ N \le max\ (measure\ L\ (space\ L))\ (measure\ N\ (space\ N))$
proof −
consider $N \notin \mathcal{P} \mid L \notin \mathcal{P}$
$\mid max\ (measure\ L\ (space\ L))\ (measure\ N\ (space\ N)) = 0\ L \in \mathcal{P}\ N \in \mathcal{P}$
$\mid max\ (measure\ L\ (space\ L))\ (measure\ N\ (space\ N)) > 0\ L \in \mathcal{P}\ N \in \mathcal{P}$
by (*metis less-max-iff-disj max.idem zero-less-measure-iff*)
then show *?thesis*
proof *cases*
assume *h*: $L \in \mathcal{P}\ N \in \mathcal{P}\ max\ (measure\ L\ (space\ L))\ (measure\ N\ (space\ N)) = 0$
interpret *L*: *finite-measure L*
using *h* **by**(*auto dest*: *inP-D*)
interpret *N*: *finite-measure N*
using *h* **by**(*auto dest*: *inP-D*)
have *measureL*:$\bigwedge A.\ measure\ L\ A = 0$
by (*metis L.bounded-measure h*(*3*) *max.absorb1 max.commute max.left-idem*

measure-nonneg)
have *measureN*:⋀*A. measure N A* = *0*
by (*metis N.bounded-measure h*(*3*) *max.absorb1 max.commute max.left-idem measure-nonneg*)
have ⋀*e. e* > *0* ⟹ *LPm L N* ≤ *e*
by(*auto intro*!: *LPm-imp-le simp*: *measureL measureN*)
thus *?thesis*
by(*simp add*: *h*(*3*) *field-le-epsilon*)
next
assume *h*:*max* (*measure L* (*space L*)) (*measure N* (*space N*)) > *0* (**is** *?a* > *0*) *L* ∈ 𝒫 *N* ∈ 𝒫
interpret *L*: *finite-measure L*
using *h* **by**(*auto dest*: *inP-D*)
interpret *N*: *finite-measure N*
using *h* **by**(*auto dest*: *inP-D*)
have ⋀*B. B* ∈ *sets* (*borel-of mtopology*) ⟹ *measure L B* ≤ *measure N* (⋃ *a*∈*B. mball a ?a*) + *?a*
using *L.bounded-measure* **by**(*auto intro*!: *add-increasing max.coboundedI1*)
moreover have ⋀*B. B* ∈ *sets* (*borel-of mtopology*) ⟹ *measure N B* ≤ *measure L* (⋃ *a*∈*B. mball a ?a*) + *?a*
using *N.bounded-measure* **by**(*auto intro*!: *add-increasing max.coboundedI2*)
ultimately show *?thesis*
by(*auto intro*!: *LPm-imp-le h*)
qed(*simp-all add*: *LPm-def max-def*)
qed

lemma *LPm-less-then*:
assumes *N* ∈ 𝒫 **and** *L* ∈ 𝒫
and *LPm N L* < *e A* ∈ *sets* (*borel-of mtopology*)
shows *measure N A* ≤ *measure L* (⋃ *a*∈*A. mball a e*) + *e measure L A* ≤ *measure N* (⋃ *a*∈*A. mball a e*) + *e*
proof −
have *sets-NL*: *sets* (*borel-of mtopology*) = *sets N sets* (*borel-of mtopology*) = *sets L*
using *assms* **by** (*auto simp*: *inP-D*)
interpret *L*: *finite-measure L*
by (*simp add*: *assms*(*2*) *inP-D*)
interpret *N*: *finite-measure N*
by (*simp add*: *assms*(*1*) *inP-D*)
have ⨅ {*e. e* > *0* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure N A ≤ *measure L* (⋃ *a*∈*A. mball a e*) + *e* ∧
measure L A ≤ *measure N* (⋃ *a*∈*A. mball a e*) + *e*)} < *e*
using *assms* **by**(*simp add*: *LPm-def*)
from *cInf-less-iff*[*THEN iffD1*,*OF LPm-ne*[*OF assms*(*1*,*2*)] *bdd-below-Levy-Prokhorov this*]
obtain *e*′ **where** *e*′:
e′ > *0* ⋀*A. A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure N A* ≤ *measure L* (⋃ *a*∈*A. mball a e*′) + *e*′
⋀*A. A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure L A* ≤ *measure N* (⋃ *a*∈*A. mball*

a e′) + *e′ e′* < *e*
  by *auto*
 have *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a e′*) + *e′*
  by(*auto intro*!: *e′ assms*)
 also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a e′*) + *e*
  using *e′* **by** *auto*
 also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*
  using *sets.sets-into-space*[*OF assms*(*4*)] *mball-subset-concentric*[*of e′ e*] *e′*
  by(*auto intro*!: *L.finite-measure-mono borel-of-open simp*: *space-borel-of sets-NL*(*2*)[*symmetric*])
 finally show *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e* **.**
 have *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e′*) + *e′*
  by(*auto intro*!: *e′ assms*)
 also have ... ≤ *measure N* (⋃ *a*∈*A*. *mball a e′*) + *e*
  using *e′* **by** *auto*
 also have ... ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e*
  using *sets.sets-into-space*[*OF assms*(*4*)] *mball-subset-concentric*[*of e′ e*] *e′*
  by(*auto intro*!: *N.finite-measure-mono borel-of-open simp*: *space-borel-of sets-NL*(*1*)[*symmetric*])
 finally show *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* **.**
qed

lemma *LPm-nonneg*:*0* ≤ *LPm N L*
 by(*auto simp*: *LPm-def le-cInf-iff*[*OF LPm-ne bdd-below-Levy-Prokhorov*])

lemma *LPm-open*: *LPm L N* = (*if L* ∈ 𝒫 ∧ *N* ∈ 𝒫 *then*
  (⨅ {*e*. *e* > *0* ∧ (∀ *A*∈{*U*. *openin mtopology U*}.
   measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball*
a e) + *e* ∧
   measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball*
a e) + *e*)})
  else 0)
proof −
 {
  assume *LN*:*L* ∈ 𝒫 *N* ∈ 𝒫
  then have *finite-measure L finite-measure N*
   and *sets-MN*[*measurable-cong*]:*sets* (*borel-of mtopology*) = *sets L sets* (*borel-of*
mtopology) = *sets N*
   by(*auto dest*: *inP-D*)
  interpret *L*: *finite-measure L* **by** *fact*
  interpret *N*: *finite-measure N* **by** *fact*
  have ⨅ {*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
    measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧ *measure N A* ≤
measure L (⋃ *a*∈*A*. *mball a e*) + *e*)} =
   ⨅ {*e*. *0* < *e* ∧ (∀ *A*. *openin mtopology A* ⟶
    measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧ *measure N A* ≤
measure L (⋃ *a*∈*A*. *mball a e*) + *e*)}
   (**is** *?lhs* = *?rhs*)
  proof(*rule order.antisym*)
   show *?rhs* ≤ *?lhs*
   using *LPm-ne*[*OF LN*] **by**(*auto intro*!: *cInf-superset-mono bdd-belowI*[**where**

m=0] *dest*: *borel-of-open*)
next
have *ball-sets*[*measurable*]: ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets L* ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets N*
by(*auto simp*: *sets-MN*[*symmetric*])
show *?lhs* ≤ *?rhs*
proof(*safe intro*!: *cInf-le-iff-less*[**where** *f=id*,*simplified*,*THEN iffD2*])
have *ne*:{*e*. *0* < *e* ∧ (∀ *A*. *openin mtopology A*
⟶ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)} ≠ {}
using *LPm-ne′*[*OF L.finite-measure-axioms N.finite-measure-axioms*] **by** *fastforce*
fix *y*
assume *y* > *?rhs*
from *cInf-lessD*[*OF ne this*] **obtain** *x* **where** *x*: *x* < *y* *0* < *x*
⋀*A*. *openin mtopology A* ⟹ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a x*) + *x*
⋀*A*. *openin mtopology A* ⟹ *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a x*) + *x*
by *auto*
define *x′* **where** *x′* ≡ *x* + (*y* − *x*) / *2*
have *x′1*: *x′* > *0* *x* < *x′*
using *x*(*1*,*2*) **by**(*auto simp*: *x′-def add-pos-pos*)
with *mball-subset-concentric*[*of x x′*] **have** *x′2*: *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a x′*) + *x′*
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a x′*) + *x′* **if** *openin mtopology A* **for** *A*
by(*auto intro*!: *order.trans*[*OF x*(*3*)[*OF that*]] *order.trans*[*OF x*(*4*)[*OF that*]]
add-mono N.finite-measure-mono L.finite-measure-mono)
show ∃ *i*∈{*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)}. *i* ≤ *y*
proof(*safe intro*!: *bexI*[**where** *x=y*])
fix *A*
assume *A*:*A* ∈ *sets* (*borel-of mtopology*)
then have [*measurable*]: *A* ∈ *sets L* *A* ∈ *sets N*
by(*auto simp*: *sets-MN*[*symmetric*])
have *measure L A* = ⨅ (*measure L* ‘ {*C*. *openin mtopology C* ∧ *A* ⊆ *C*})
by(*simp add*: *L.outer-regularD*[*OF L.outer-regular′*[*OF metrizable-space-mtopology sets-MN*(*1*)]]])
also have ... ≤ ⨅ {*measure N* (⋃ *c*∈*C*. *mball c x′*) + *x′* |*C*. *openin mtopology C* ∧ *A* ⊆ *C*}
using *sets.sets-into-space*[*OF A*]
by(*auto intro*!: *cInf-mono x′2 bdd-belowI*[**where** *m=0*] *simp*: *space-borel-of*)
also have ... ≤ *measure N* (⋃ *a*∈(⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*)). *mball a x′*) + *x′*
proof(*safe intro*!: *cInf-lower bdd-belowI*[**where** *m=0*])

have *A* ⊆ (⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*))
using *x*(*1*) *sets.sets-into-space*[*OF A*] **by**(*fastforce simp*: *space-borel-of*)
thus ∃ *C*. *measure N* (⋃ *b*∈(⋃ *a*∈*A*. *mball a* ((*y* − *x*) / *2*)). *mball b x*′) + *x*′
= *measure N* (⋃ *c*∈*C*. *mball c x*′) + *x*′ ∧ *openin mtopology C* ∧ *A* ⊆ *C*
by(*auto intro*!: *exI*[**where** *x*=⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*)])
qed(*use measure-nonneg x*′*1* **in** *auto*)
also have ... ≤ *measure N* (⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2* + *x*′)) + *x*′
using *nbh-add*[*of x*′ (*y*−*x*)/*2 A*] **by**(*auto intro*!: *N.finite-measure-mono*)
also have ... = *measure N* (⋃ *a*∈*A*. *mball a y*) + *x*′
by(*auto simp*: *x*′-*def*)
also have ... ≤ *measure N* (⋃ *a*∈*A*. *mball a y*) + *y*
using *x*(*1*,*2*)
by(*auto simp*: *x*′-*def intro*!: *order.trans*[*OF le-add-same-cancel1*[*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a y*) + *y* **.**
have *measure N A* = ⨅ (*measure N* ‘ {*C*. *openin mtopology C* ∧ *A* ⊆ *C*})
by(*simp add*: *N.outer-regularD*[*OF N.outer-regular*′[*OF metrizable-space-mtopology sets-MN*(*2*)]]])
also have ... ≤ ⨅ {*measure L* (⋃ *c*∈*C*. *mball c x*′) + *x*′ |*C*. *openin mtopology C* ∧ *A* ⊆ *C*}
using *sets.sets-into-space*[*OF A*]
by(*auto intro*!: *cInf-mono x*′*2 bdd-belowI*[**where** *m*=*0*] *simp*: *space-borel-of*)
also have ... ≤ *measure L* (⋃ *a*∈(⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*)). *mball a x*′) + *x*′
proof(*safe intro*!: *cInf-lower bdd-belowI*[**where** *m*=*0*])
have *A* ⊆ (⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*))
using *x*(*1*) *sets.sets-into-space*[*OF A*] **by**(*fastforce simp*: *space-borel-of*)
thus ∃ *C*. *measure L* (⋃ *b*∈⋃ *a*∈*A*. *mball a* ((*y* − *x*) / *2*). *mball b x*′) + *x*′
= *measure L* (⋃ *c*∈*C*. *mball c x*′) + *x*′ ∧ *openin mtopology C* ∧ *A* ⊆ *C*
by(*auto intro*!: *exI*[**where** *x*=⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*)])
qed(*use measure-nonneg x*′*1* **in** *auto*)
also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2* + *x*′)) + *x*′
using *nbh-add*[*of x*′ (*y*−*x*)/*2 A*] **by**(*auto intro*!: *L.finite-measure-mono*)
also have ... = *measure L* (⋃ *a*∈*A*. *mball a y*) + *x*′
by(*auto simp*: *x*′-*def*)
also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a y*) + *y*
using *x*(*1*,*2*)
by(*auto simp*: *x*′-*def intro*!: *order.trans*[*OF le-add-same-cancel1*[*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a y*) + *y* **.**
qed(*use x* **in** *auto*)
qed(*insert LPm-ne*[*OF LN*], *auto intro*!: *bdd-belowI*[**where** *m*=*0*])
qed
}
thus *?thesis*

by (*auto simp*: *LPm-def*)
qed

lemma *LPm-closed*: *LPm L N* = (*if L* ∈ 𝒫 ∧ *N* ∈ 𝒫 *then*
(⨅ {*e*. *e* > *0* ∧ (∀ *A*∈{*U*. *closedin mtopology U*}.
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)})
else 0)
proof −
{
assume *LN*:*L* ∈ 𝒫 *N* ∈ 𝒫
then have *finite-measure L finite-measure N*
and *sets-MN*[*measurable-cong*]: *sets* (*borel-of mtopology*) = *sets L sets* (*borel-of mtopology*) = *sets N*
by(*auto dest*: *inP-D*)
interpret *L*: *finite-measure L* **by** *fact*
interpret *N*: *finite-measure N* **by** *fact*
have ⨅ {*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)}
= ⨅ {*e*. *0* < *e* ∧ (∀ *A*. *closedin mtopology A* ⟶
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)} (**is** *?lhs* = *?rhs*)
proof(*rule order.antisym*)
show *?rhs* ≤ *?lhs*
using *LPm-ne*[*OF LN*] **by**(*auto intro*!: *cInf-superset-mono bdd-belowI*[**where** *m*=*0*] *dest*: *borel-of-closed*)
next
have *ball-sets*[*measurable*]: ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets L* ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets N*
by(*auto simp*: *sets-MN*[*symmetric*])
show *?lhs* ≤ *?rhs*
proof(*safe intro*!: *cInf-le-iff-less*[**where** *f*=*id*,*simplified*,*THEN iffD2*])
have *ne*:{*e*. *0* < *e* ∧ (∀ *A*. *closedin mtopology A* ⟶ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e*
∧ *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)} ≠ {}
using *LPm-ne′*[*OF L.finite-measure-axioms N.finite-measure-axioms*] **by** *fastforce*
fix *y*
assume *y* > *?rhs*
from *cInf-lessD*[*OF ne this*] **obtain** *x* **where** *x*: *x* < *y* *0* < *x*
⋀*A*. *closedin mtopology A* ⟹ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a x*) + *x*
⋀*A*. *closedin mtopology A* ⟹ *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a x*) + *x*

by *auto*
define *x′* **where** *x′ ≡ x + (y − x) / 2*
have *x′1*: *x′ > 0 x < x′*
using *x(1,2)* **by**(*auto simp*: *x′-def add-pos-pos*)
with *mball-subset-concentric[of x x′]*
have *x′2*: *measure L A ≤ measure N (⋃ a∈A. mball a x′) + x′ measure N A ≤ measure L (⋃ a∈A. mball a x′) + x′*
if *closedin mtopology A* **for** *A*
by(*auto intro*!: *order.trans[OF x(3)[OF that]] order.trans[OF x(4)[OF that]]*
add-mono N.finite-measure-mono L.finite-measure-mono)
show *∃ i∈{e. 0 < e ∧ (∀ A∈sets (borel-of mtopology). measure L A ≤ measure N (⋃ a∈A. mball a e) + e ∧*
measure N A ≤ measure L (⋃ a∈A. mball a e) + e)}. i ≤ y
proof(*safe intro*!: *bexI*[**where** *x=y*])
fix *A*
assume *A*:*A ∈ sets (borel-of mtopology)*
then have [*measurable*]: *A ∈ sets L A ∈ sets N*
by(*auto simp*: *sets-MN[symmetric]*)
have *measure L A = (⨆ (measure L ‘ {C. closedin mtopology C ∧ C ⊆ A}))*
by(*simp add*: *L.inner-regularD[OF L.inner-regular′[OF metrizable-space-mtopology sets-MN(1)]]*)
also have ... *≤ (⨆ {measure N (⋃ c∈C. mball c x′) + x′ |C. closedin mtopology C ∧ C ⊆ A})*
using *sets.sets-into-space[OF A]*
by(*auto intro*!: *cSup-mono x′2 bdd-aboveI*[**where** *M=measure N (space N) + x′*] *N.bounded-measure*
simp: *space-borel-of*)
also have ... *≤ measure N (⋃ a∈(⋃ a∈A. mball a ((y−x)/2)). mball a x′) + x′*
proof(*safe intro*!: *cSup-le-iff[THEN iffD2] bdd-aboveI*[**where** *M=measure N (space N) + x′*])
fix *C*
assume *C ⊆ A*
then have *(⋃ c∈C. mball c x′) ⊆ (⋃ b∈⋃ a∈A. mball a ((y − x) / 2). mball b x′)*
using *x′1(2) x′-def* **by** *fastforce*
thus *measure N (⋃ c∈C. mball c x′) + x′ ≤ measure N (⋃ b∈⋃ a∈A. mball a ((y − x) / 2). mball b x′) + x′*
by (*metis N.finite-measure-mono add.commute add-le-cancel-left ball-sets(2)*)
qed(*auto intro*!: *N.bounded-measure*)
also have ... *≤ measure N (⋃ a∈A. mball a ((y−x)/2 + x′)) + x′*
using *nbh-add[of x′ (y−x)/2 A]* **by**(*auto intro*!: *N.finite-measure-mono*)
also have ... *= measure N (⋃ a∈A. mball a y) + x′*
by(*auto simp*: *x′-def*)
also have ... *≤ measure N (⋃ a∈A. mball a y) + y*

using *x*(*1*,*2*)
by(*auto simp*: *x′-def intro*!: *order.trans*[*OF le-add-same-cancel1* [*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a y*) + *y* **.**
have *measure N A* = ⨆ (*measure N* ‘ {*C*. *closedin mtopology C* ∧ *C* ⊆ *A*})
by(*simp add*: *N.inner-regularD*[*OF N.inner-regular′*[*OF metrizable-space-mtopology sets-MN*(*2*)]])
also have ... ≤ ⨆ {*measure L* (⋃ *c*∈*C*. *mball c x′*) + *x′* |*C*. *closedin mtopology C* ∧ *C* ⊆ *A*}
using *sets.sets-into-space*[*OF A*]
by(*auto intro*!: *cSup-mono x′2 bdd-aboveI*[**where** *M*=*measure L* (*space L*) + *x′*] *L.bounded-measure*
simp: *space-borel-of*)
also have ... ≤ *measure L* (⋃ *a*∈(⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2*)). *mball a x′*) + *x′*
proof(*safe intro*!: *cSup-le-iff* [*THEN iffD2*] *bdd-aboveI*[**where** *M*=*measure L* (*space L*) + *x′*])
fix *C*
assume *C* ⊆ *A*
then have (⋃ *c*∈*C*. *mball c x′*) ⊆ (⋃ *b*∈⋃ *a*∈*A*. *mball a* ((*y* − *x*) / *2*). *mball b x′*)
using *x′1*(*2*) *x′-def* **by** *fastforce*
thus *measure L* (⋃ *c*∈*C*. *mball c x′*) + *x′* ≤ *measure L* (⋃ *b*∈⋃ *a*∈*A*. *mball a* ((*y* − *x*) / *2*). *mball b x′*) + *x′*
by (*metis L.finite-measure-mono add.commute add-le-cancel-left ball-sets*(*1*))
qed(*auto intro*!: *L.bounded-measure*)
also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a* ((*y*−*x*)/*2* + *x′*)) + *x′*
using *nbh-add*[*of x′* (*y*−*x*)/*2 A*] **by**(*auto intro*!: *L.finite-measure-mono*)
also have ... = *measure L* (⋃ *a*∈*A*. *mball a y*) + *x′*
by(*auto simp*: *x′-def*)
also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a y*) + *y*
using *x*(*1*,*2*)
by(*auto simp*: *x′-def intro*!: *order.trans*[*OF le-add-same-cancel1* [*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a y*) + *y* **.**
qed(*use x* **in** *auto*)
qed(*insert LPm-ne*[*OF LN*], *auto intro*!: *bdd-belowI*[**where** *m*=*0*])
qed
}
thus *?thesis*
by (*auto simp*: *LPm-def*)
qed

lemma *LPm-compact*:
assumes *separable-space mtopology mcomplete*
shows *LPm L N* = (*if L* ∈ 𝒫 ∧ *N* ∈ 𝒫 *then*
(⨅ {*e*. *e* > *0* ∧ (∀ *A*∈{*U*. *compactin mtopology U*}.

measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*)
+ *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*)
+ *e*)})
else 0)
proof −
{
assume *LN*:*L* ∈ 𝒫 *N* ∈ 𝒫
then have *finite-measure L finite-measure N*
and *sets-MN*[*measurable-cong*]: *sets* (*borel-of mtopology*) = *sets L sets* (*borel-of mtopology*) = *sets N*
by(*auto dest*: *inP-D*)
interpret *L*: *finite-measure L* **by** *fact*
interpret *N*: *finite-measure N* **by** *fact*
have *measureL*:*A* ∈ *sets L* ⟹ *measure L A* = (⨆ *K*∈{*K*. *compactin mtopology K* ∧ *K* ⊆ *A*}. *measure L K*)
and *measureN*: *A* ∈ *sets N* ⟹ *measure N A* = (⨆ *K*∈{*K*. *compactin mtopology K* ∧ *K* ⊆ *A*}. *measure N K*) **for** *A*
by(*auto intro*!: *inner-regular'' L.tight-on-Polish N.tight-on-Polish Polish-space-mtopology assms*
simp: *sets-MN*[*symmetric*] *metrizable-space-mtopology*)
have ⨅ {*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)}
= ⨅ {*e*. *0* < *e* ∧ (∀ *A*. *compactin mtopology A* ⟶
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)}
(**is** *?lhs* = *?rhs*)
proof(*rule order.antisym*)
show *?rhs* ≤ *?lhs*
using *LPm-ne*[*OF LN*] **by**(*auto intro*!: *cInf-superset-mono bdd-belowI*[**where** *m*=*0*]
dest: *borel-of-compact*[*OF Hausdorff-space-mtopology*])
next
have *ball-sets*[*measurable*]: ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets L* ⋀*A e*. (⋃ *a*∈*A*. *mball a e*) ∈ *sets N*
by(*auto simp*: *sets-MN*[*symmetric*])
show *?lhs* ≤ *?rhs*
proof(*safe intro*!: *cInf-le-iff-less*[**where** *f*=*id*,*simplified*,*THEN iffD2*])
have *ne*:{*e*. *0* < *e* ∧ (∀ *A*. *compactin mtopology A* ⟶
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) +
e)} ≠ {}
using *LPm-ne'*[*OF L.finite-measure-axioms N.finite-measure-axioms*] **by** *fastforce*
fix *y*
assume *y* > *?rhs*
from *cInf-lessD*[*OF ne this*] **obtain** *x* **where** *x*: *x* < *y* *0* < *x*
⋀*A*. *compactin mtopology A* ⟹ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball*

a x) + x
⋀A. compactin mtopology A ⟹ measure N A ≤ measure L (⋃ a∈A. mball a x) + x
by auto
define x′ where x′ ≡ x + (y − x) / 2
have x′1: x′ > 0 x < x′
using x(1,2) by(auto simp: x′-def add-pos-pos)
with mball-subset-concentric[of x x′]
have x′2: measure L A ≤ measure N (⋃ a∈A. mball a x′) + x′ measure N A ≤ measure L (⋃ a∈A. mball a x′) + x′
if compactin mtopology A for A
by(auto intro!: order.trans[OF x(3)[OF that]] order.trans[OF x(4)[OF that]]
add-mono N.finite-measure-mono L.finite-measure-mono)
show ∃ i∈{e. 0 < e ∧ (∀ A∈sets (borel-of mtopology). measure L A ≤ measure N (⋃ a∈A. mball a e) + e ∧
measure N A ≤ measure L (⋃ a∈A. mball a e) + e)}. i ≤ y
proof(safe intro!: bexI[where x=y])
fix A
assume A:A ∈ sets (borel-of mtopology)
then have [measurable]: A ∈ sets L A ∈ sets N
by(auto simp: sets-MN[symmetric])
have measure L A = (⨆ (measure L ‘ {C. compactin mtopology C ∧ C ⊆ A}))
by(simp add: measureL)
also have ... ≤ (⨆ {measure N (⋃ c∈C. mball c x′) + x′ |C. compactin mtopology C ∧ C ⊆ A})
using sets.sets-into-space[OF A]
by(auto intro!: cSup-mono x′2 bdd-aboveI[where M=measure N (space N) + x′] N.bounded-measure
simp: space-borel-of)
also have ... ≤ measure N (⋃ a∈(⋃ a∈A. mball a ((y−x)/2)). mball a x′) + x′
proof(safe intro!: cSup-le-iff[THEN iffD2] bdd-aboveI[where M=measure N (space N) + x′])
fix C
assume C ⊆ A
then have (⋃ c∈C. mball c x′) ⊆ (⋃ b∈⋃ a∈A. mball a ((y − x) / 2). mball b x′)
using x′1(2) x′-def by fastforce
thus measure N (⋃ c∈C. mball c x′) + x′ ≤ measure N (⋃ b∈⋃ a∈A. mball a ((y − x) / 2). mball b x′) + x′
by (metis N.finite-measure-mono add.commute add-le-cancel-left ball-sets(2))
qed(auto intro!: N.bounded-measure)
also have ... ≤ measure N (⋃ a∈A. mball a ((y−x)/2 + x′)) + x′
using nbh-add[of x′ (y−x)/2 A] by(auto intro!: N.finite-measure-mono)
also have ... = measure N (⋃ a∈A. mball a y) + x′

by(*auto simp*: *x′-def*)
also have ... ≤ *measure N* (⋃*a*∈*A*. *mball a y*) + *y*
using *x*(*1*,*2*)
by(*auto simp*: *x′-def intro*!: *order.trans*[*OF le-add-same-cancel1*[*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure L A* ≤ *measure N* (⋃*a*∈*A*. *mball a y*) + *y* **.**
have *measure N A* = ⨆ (*measure N* ‘ {*C*. *compactin mtopology C* ∧ *C* ⊆ *A*})
by(*simp add*: *measureN*)
also have ... ≤ ⨆ {*measure L* (⋃*c*∈*C*. *mball c x′*) + *x′* |*C*. *compactin mtopology C* ∧ *C* ⊆ *A*}
using *sets.sets-into-space*[*OF A*]
by(*auto intro*!: *cSup-mono x′2 bdd-aboveI*[**where** *M*=*measure L* (*space L*) + *x′*] *L.bounded-measure*
simp: *space-borel-of*)
also have ... ≤ *measure L* (⋃*a*∈(⋃*a*∈*A*. *mball a* ((*y*−*x*)/*2*)). *mball a x′*) + *x′*
proof(*safe intro*!: *cSup-le-iff*[*THEN iffD2*] *bdd-aboveI*[**where** *M*=*measure L* (*space L*) + *x′*])
fix *C*
assume *C* ⊆ *A*
then have (⋃*c*∈*C*. *mball c x′*) ⊆ (⋃*b*∈⋃*a*∈*A*. *mball a* ((*y* − *x*) / *2*). *mball b x′*)
using *x′1*(*2*) *x′-def* **by** *fastforce*
thus *measure L* (⋃*c*∈*C*. *mball c x′*) + *x′* ≤ *measure L* (⋃*b*∈⋃*a*∈*A*. *mball a* ((*y* − *x*) / *2*). *mball b x′*) + *x′*
by (*metis L.finite-measure-mono add.commute add-le-cancel-left ball-sets*(*1*))
qed(*auto intro*!: *L.bounded-measure*)
also have ... ≤ *measure L* (⋃*a*∈*A*. *mball a* ((*y*−*x*)/*2* + *x′*)) + *x′*
using *nbh-add*[*of x′* (*y*−*x*)/*2 A*] **by**(*auto intro*!: *L.finite-measure-mono*)
also have ... = *measure L* (⋃*a*∈*A*. *mball a y*) + *x′*
by(*auto simp*: *x′-def*)
also have ... ≤ *measure L* (⋃*a*∈*A*. *mball a y*) + *y*
using *x*(*1*,*2*)
by(*auto simp*: *x′-def intro*!: *order.trans*[*OF le-add-same-cancel1*[*of x*+(*y*−*x*)/*2* (*y*−*x*)/*2*,*THEN iffD2*]])
finally show *measure N A* ≤ *measure L* (⋃*a*∈*A*. *mball a y*) + *y* **.**
qed(*use x* **in** *auto*)
qed(*insert LPm-ne*[*OF LN*], *auto intro*!: *bdd-belowI*[**where** *m*=*0*])
qed
}
thus *?thesis*
by (*auto simp*: *LPm-def*)
qed

sublocale *LPm*: *Metric-space* 𝒫 *LPm*
proof
show *0* ≤ *LPm M N* **for** *M N*

by(*rule LPm-nonneg*)
next
fix *L N*
assume *MN*:*L* ∈ 𝒫 *N* ∈ 𝒫
interpret *L*: *finite-measure L*
by(*rule inP-D*(*1*)[*OF MN*(*1*)])
interpret *N*: *finite-measure N*
by(*rule inP-D*(*1*)[*OF MN*(*2*)])
show *LPm L N* = *0* ⟷ *L* = *N*
proof *safe*
have [*simp*]:{*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*). *measure N A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e*)} = {*0*<..}
proof *safe*
fix *e* :: *real* **and** *A*
assume *h'*:*e* > *0* *A* ∈ *sets* (*borel-of mtopology*)
show *measure N A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e*
using *nbh-sets*[*of e A*] *inP-D*(*2*)[*OF MN*(*2*)] *sets.sets-into-space*[*OF h'*(*2*)] *h'*(*1*)
by(*auto simp*: *space-borel-of intro*!: *order.trans*[*OF N.finite-measure-mono*[*OF nbh-subset*[*of A e*]]])
qed
show *LPm N N* = *0*
by (*simp add*: *LPm-def*)
next
assume *LPm L N* = *0*
then have *h*:⋀*e'*. *e'* > *0* ⟹
∃ *a*∈{*e*. *0* < *e* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e*)}. *a* < *e'*
using *cInf-le-iff*[*OF LPm-ne*[*OF MN*] *bdd-below-Levy-Prokhorov*] **by** (*auto simp*: *MN LPm-def*)
show *L* = *N*
proof(*rule measure-eqI-generator-eq*[**where** *E*={*U*. *closedin mtopology U*} **and** *A*=λ*i*. *M* **and** Ω=*M*])
show *Int-stable* {*U*. *closedin mtopology U*}
by(*auto simp*: *Int-stable-def*)
next
show {*U*. *closedin mtopology U*} ⊆ *Pow M*
using *closedin-metric2* **by** *auto*
next
show ⋀*X*. *X* ∈ {*U*. *closedin mtopology U*} ⟹ *emeasure L X* = *emeasure N X*
proof *safe*
fix *U*
assume *closedin mtopology U*
then have *US*: *U* ⊆ *M*
by (*simp add*: *closedin-def*)
consider *U* = {} | *U* ≠ {} **by** *auto*
then have *measure L U* = *measure N U*

proof *cases*
case *U*:*2*
define *an*
where *an* ≡ *rec-nat* (*SOME e. 0 < e* ∧ *e < 1 / Suc 0*
∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a*
e) + *e*
∧ *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a*
e) + *e*))
(λ*n an*. *SOME e. 0 < e* ∧ *e < an* ∧ *e < 1 / Suc* (*Suc n*)
∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball*
a e) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball*
a e) + *e*))
have *an-simp*: *an 0* = (*SOME e. 0 < e* ∧ *e < 1 / Suc 0*
∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a*
e) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a*
e) + *e*))
⋀*n*. *an* (*Suc n*) = (*SOME e. 0 < e* ∧ *e <* (*an n*) ∧ *e < 1 / Suc*
(*Suc n*) ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball*
a e) + *e* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball*
a e) + *e*))
by(*simp-all add*: *an-def*)
have ∗:*an 0 > 0* ∧ *an 0 < 1 / Suc 0* ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*an 0*)) + (*an 0*) ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*an 0*)) + (*an 0*))
by(*simp add*: *an-simp*) (*rule someI-ex*,*use h*[*of 1*] **in** *auto*)
moreover have ∗∗:*an n > 0* **for** *n*
proof(*induction n*)
case *ih*:(*Suc n*)
have *an* (*Suc n*) *> 0* ∧ *an* (*Suc n*) *< an n* ∧ *an* (*Suc n*) *< 1 / Suc*
(*Suc n*) ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*an* (*Suc n*))) +
(*an* (*Suc n*)) ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*an* (*Suc n*))) +
(*an* (*Suc n*)))
by(*simp add*: *an-simp*,*rule someI-ex*) (*use h*[*of min* (*an n*) (*1 / Suc*
(*Suc n*))] *ih* **in** *auto*)
thus *?case*
by *auto*
qed(*use* ∗ **in** *auto*)

moreover have *an* (*Suc n*) > *0* ∧ *an* (*Suc n*) < *an n* ∧ *an* (*Suc n*) < *1* / *Suc* (*Suc n*) ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*an* (*Suc n*))) + (*an* (*Suc n*)) ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*an* (*Suc n*))) + (*an* (*Suc n*))) **for** *n*
by(*simp add*: *an-simp*,*rule someI-ex*) (*use h*[*of min* (*an n*) (*1* / *Suc* (*Suc n*))] ∗∗ **in** *auto*)
ultimately have *an n* > *0* ∧ *decseq an* ∧ *an n* < *1* / *Suc n* ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*an n*)) + (*an n*) ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*an n*)) + (*an n*)) **for** *n*
by(*cases n*) (*auto intro*!: *decseq-SucI order.strict-implies-order*)
hence *an*:⋀*n*. *an n* > *0* *decseq an* ⋀*n*. *an n* < *1* / *Suc n*
⋀*n A*. *A*∈*sets* (*borel-of mtopology*) ⟹ *measure L A* ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*an n*)) + *an n*
⋀*n A*. *A*∈*sets* (*borel-of mtopology*) ⟹ *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*an n*)) + *an n*
by *auto*
hence *an-lim*: *an* ⟶ *0*
by(*auto intro*!: *LIMSEQ-norm-0 simp*: *less-eq-real-def*)
have *1*:*U* ∈ *sets* (*borel-of mtopology*)
by (*simp add*: ‹*closedin mtopology U*› *borel-of-closed*)
have *Uint*:(⋂ *i*. ⋃ *a*∈*U*. *mball a* (*an i*)) = *U*
by(*simp add*: *nbh-Inter-closure-of*[*OF U US an*(*1*,*2*) *an-lim*] *closure-of-closedin*[*OF* ‹*closedin mtopology U*›])
have (λ*n*. *measure L* (⋃ *a*∈*U*. *mball a* (*an n*))) ⟶ *measure L* (⋂ *i*. ⋃ *a*∈*U*. *mball a* (*an i*))
(λ*n*. *measure N* (⋃ *a*∈*U*. *mball a* (*an n*))) ⟶ *measure N* (⋂ *i*. ⋃ *a*∈*U*. *mball a* (*an i*))
by(*auto intro*!: *L.finite-Lim-measure-decseq*[*OF - nbh-decseq*[*OF an*(*2*)]]
N.finite-Lim-measure-decseq[*OF - nbh-decseq*[*OF an*(*2*)]] *simp*: *MN*)
hence *MN-lim*:(λ*n*. *measure L* (⋃ *a*∈*U*. *mball a* (*an n*)) + *an n*) ⟶ *measure L U*
(λ*n*. *measure N* (⋃ *a*∈*U*. *mball a* (*an n*)) + *an n*) ⟶ *measure N U*
by(*auto simp add*: *Uint intro*!: *tendsto-add*[*OF - an-lim*,*simplified*])
show *?thesis*
proof(*rule order.antisym*)
show *measure L U* ≤ *measure N U*
by(*rule Lim-bounded2*[*OF MN-lim*(*2*)],*auto simp*: *an 1*)
next
show *measure N U* ≤ *measure L U*
by(*rule Lim-bounded2*[*OF MN-lim*(*1*)],*auto simp*: *an 1*)
qed
qed *simp*

thus *emeasure L U* = *emeasure N U*
by (*simp add*: *L.emeasure-eq-measure N.emeasure-eq-measure*)
qed
next
show *range* (λ*i*. *M*) ⊆ {*U*. *closedin mtopology U*}
by *simp*
qed (*simp-all add*: *MN sets-borel-of-closed inP-D*(*2*))
qed
next
fix *M N L*
assume *MNL*[*simp*]: *M* ∈ 𝒫 *N* ∈ 𝒫 *L* ∈ 𝒫
interpret *M*: *finite-measure M*
by(*rule inP-D*(*1*)[*OF MNL*(*1*)])
interpret *N*: *finite-measure N*
by(*rule inP-D*(*1*)[*OF MNL*(*2*)])
interpret *L*: *finite-measure L*
by(*rule inP-D*(*1*)[*OF MNL*(*3*)])
have *ne*:{*e1* + *e2* |*e1 e2*. *0* < *e1* ∧ *0* < *e2* ∧
(∀ *A*∈*sets* (*borel-of mtopology*).
measure M A ≤ *measure N* (⋃ *a*∈*A*. *mball a e1*) + *e1* ∧
measure N A ≤ *measure M* (⋃ *a*∈*A*. *mball a e1*) + *e1* ∧
measure N A ≤ *measure L* (⋃ *a*∈*A*. *mball a e2*) + *e2* ∧
measure L A ≤ *measure N* (⋃ *a*∈*A*. *mball a e2*) + *e2*)} ≠ {}
(**is** {*e1* + *e2* | *e1 e2*. *0* < *e1* ∧ *0* < *e2* ∧ *?P e1 e2*} ≠ {})
using *N.bounded-measure M.bounded-measure L.bounded-measure*
by(*auto intro*!: *exI*[**where** *x*=*max 1* (*max* (*measure M* (*space M*)) (*max* (*measure L* (*space L*)) (*measure N* (*space N*))))]
add-increasing[*OF measure-nonneg*] *simp*: *le-max-iff-disj*)
show *LPm M L* ≤ *LPm M N* + *LPm N L* (**is** *?lhs* ≤ *?rhs*)
proof −
have *?lhs* = ⨅ {*e*. *e* > *0* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure M A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure L A ≤ *measure M* (⋃ *a*∈*A*. *mball a e*) + *e*)}
by(*auto simp*: *LPm-def*)
also have ... ≤ ⨅ {*e1* + *e2* |*e1 e2*. *0* < *e1* ∧ *0* < *e2* ∧ *?P e1 e2*} (**is** - ≤ *Inf ?B*)
proof(*rule cInf-superset-mono*)
show *?B* ⊆ {*e*. *e* > *0* ∧ (∀ *A*∈*sets* (*borel-of mtopology*).
measure M A ≤ *measure L* (⋃ *a*∈*A*. *mball a e*) + *e* ∧
measure L A ≤ *measure M* (⋃ *a*∈*A*. *mball a e*) + *e*)}
proof *safe*
fix *e1 e2 A*
assume *?P e1 e2*
and *A*[*measurable*]: *A* ∈ *sets* (*borel-of mtopology*)
then have *mA*:
⋀*A*. *A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure M A* ≤ *measure N* (⋃ *a*∈*A*. *mball a e1*) + *e1*
⋀*A*. *A* ∈ *sets* (*borel-of mtopology*) ⟹ *measure N A* ≤ *measure M* (⋃ *a*∈*A*. *mball a e1*) + *e1*

⋀A. A ∈ sets (borel-of mtopology) ⟹ measure N A ≤ measure L (⋃a∈A. mball a e2) + e2
⋀A. A ∈ sets (borel-of mtopology) ⟹ measure L A ≤ measure N (⋃a∈A. mball a e2) + e2
by *auto*
show *measure M A ≤ measure L (⋃a∈A. mball a (e1 + e2)) + (e1 + e2)*
proof −
have *measure M A ≤ measure N (⋃a∈A. mball a e1) + e1*
by(*simp add*: *mA*)
also have ... ≤ *measure L (⋃a∈(⋃a∈A. mball a e1). mball a e2) + e2 + e1*
by(*simp add*: *mA*(*3*)[*of* ⋃a∈A. *mball a e1*,*simplified*])
also have ... ≤ *measure L (⋃a∈A. mball a (e1 + e2)) + e2 + e1*
by(*simp add*: *L.finite-measure-mono*[*OF nbh-add*,*simplified*])
finally show *?thesis*
by *simp*
qed
show *measure L A ≤ measure M (⋃a∈A. mball a (e1 + e2)) + (e1 + e2)*
proof −
have *measure L A ≤ measure N (⋃a∈A. mball a e2) + e2*
by(*simp add*: *mA*)
also have ... ≤ *measure M (⋃a∈(⋃a∈A. mball a e2). mball a e1) + e1 + e2*
by(*simp add*: *mA*(*2*)[*of* ⋃a∈A. *mball a e2*,*simplified*])
also have ... ≤ *measure M (⋃a∈A. mball a (e1 + e2)) + e1 + e2*
by(*simp add*: *M.finite-measure-mono*[*OF nbh-add*,*simplified*] *add.commute*[*of e1*])
finally show *?thesis*
by *simp*
qed
qed *simp*
qed (*use ne bdd-below-Levy-Prokhorov* **in** *auto*)
also have ... ≤ *?rhs*
proof(*rule cInf-le-iff-less*[**where** *f=id*,*simplified*,*THEN iffD2*])
show ∀ y>LPm M N + LPm N L. ∃ i∈{e1 + e2 |e1 e2. 0 < e1 ∧ 0 < e2 ∧ ?P e1 e2}. i ≤ y
proof *safe*
fix *e*
assume *h*:*LPm M N + LPm N L < e*
define *e′* **where** *e′ ≡ (e − (LPm M N + LPm N L)) / 2*
have *e′*[*arith*]: *e′ > 0*
using *h* **by**(*auto simp*: *e′-def*)
have
⋀y. y>LPm M N ⟹ ∃ i∈{e. 0 < e ∧ (∀ A∈sets (borel-of mtopology). measure M A ≤ measure N (⋃a∈A. mball a e) + e ∧ measure N A ≤ measure M (⋃a∈A. mball a e) + e)}. i ≤ y
⋀y. y>LPm N L ⟹ ∃ i∈{e. 0 < e ∧ (∀ A∈sets (borel-of mtopology).

measure N A ≤ measure L (⋃ a∈A. mball a
e) + e ∧
measure L A ≤ measure N (⋃ a∈A. mball a
e) + e)}. i ≤ y
using *cInf-le-iff-less*[**where** *f=id,simplified,OF LPm-ne*[*OF MNL(2,3)*],*of
LPm N L*]
cInf-le-iff-less[**where** *f=id,simplified,OF LPm-ne*[*OF MNL(1,2)*],*of LPm
M N*]
by(*simp-all add*: *LPm-def bdd-below-Levy-Prokhorov*)
from *this(1)*[*of LPm M N + e'*] *this(2)*[*of LPm N L + e'*] **obtain** *emn enl*
where *emn*: *emn > 0 emn ≤ LPm M N + e'*
⋀A. A ∈ sets (borel-of mtopology) ⟹ measure M A ≤ measure N
(⋃ a∈A. mball a emn) + emn
⋀A. A ∈ sets (borel-of mtopology) ⟹ measure N A ≤ measure M
(⋃ a∈A. mball a emn) + emn
and *enl*: *enl > 0 enl ≤ LPm N L + e'*
⋀A. A ∈ sets (borel-of mtopology) ⟹ measure N A ≤ measure L (⋃ a∈A.
mball a enl) + enl
⋀A. A ∈ sets (borel-of mtopology) ⟹ measure L A ≤ measure N (⋃ a∈A.
mball a enl) + enl
by *auto*
hence *emn + enl ≤ e*
by(*auto intro*!: *order.trans*[*of emn + enl LPm M N + e' + (LPm N L +
e') e*])
(*auto simp*: *e'-def diff-divide-distrib*)
show ∃ i∈{e1 + e2 |e1 e2. 0 < e1 ∧ 0 < e2 ∧ ?P e1 e2}. i ≤ e
apply(*rule bexI*[**where** *x=emn + enl*])
apply *fact*
apply *standard*
apply(*rule exI*[**where** *x=emn*])
apply(*rule exI*[**where** *x=enl*])
apply(*use emn enl* **in** *auto*)
done
qed
qed(*insert ne,auto intro*!: *bdd-belowI*[*of - 0*])
finally show *?thesis* **.**
qed
qed(*simp add*: *LPm-def*, *meson*)

4.2 Convervence and Weak Convergence

lemma *converge-imp-mweak-conv*:
assumes *limitin LPm.mtopology Ni N F*
shows *mweak-conv Ni N F*
proof(*cases F = ⊥*)
assume *F*: *F ≠ ⊥*
have *h*: *N ∈ 𝒫 ((λn. LPm (Ni n) N) ⟶ 0) F ∀_F i in F. Ni i ∈ 𝒫*
using *LPm.limitin-metric-dist-null assms(1)* **by** *auto*
interpret *N*: *finite-measure N*

using *h* **by**(*auto simp*: *inP-D*)
interpret *mweak-conv-fin M d Ni N*
by(*auto intro*!: *h inP-mweak-conv-fin assms*)
show *?thesis*
unfolding *mweak-conv-eq2*
proof *safe*
show ((λ*n*. *measure* (*Ni n*) *M*) ⟶ *measure N M*) *F*
unfolding *tendsto-iff dist-real-def*
proof *safe*
fix *r* :: *real*
assume *r*: *0* < *r*
from *half-gt-zero*[*OF this*] *h*(*2*)
have *1*:∀$_F$ *n in F*. *LPm* (*Ni n*) *N* < *r* / *2*
unfolding *tendsto-iff dist-real-def LPm.nonneg* **by** *fastforce*
show ∀$_F$ *n in F*. |*measure* (*Ni n*) *M* − *measure N M*| < *r*
proof(*safe intro*!: *eventually-mono*[*OF eventually-conj*[*OF h*(*3*) *1*]])
fix *n*
assume *n*: *LPm* (*Ni n*) *N* < *r* / *2 Ni n* ∈ 𝒫
have [*simp*]:(⋃*a*∈*M*. *mball a* (*r* / *2*)) = *M*
using *r* **by** *auto*
have [*measurable*]: *M* ∈ *sets* (*borel-of mtopology*)
by(*auto intro*!: *borel-of-open*)
have *measure* (*Ni n*) *M* ≤ *measure N M* + *r* / *2 measure N M* ≤ *measure* (*Ni n*) *M* + *r* / *2*
using *LPm-less-then*[*OF* - - *n*(*1*),*of M*] *h*(*1*) *n*(*2*) **by** *auto*
hence |*measure* (*Ni n*) *M* −*measure N M*| ≤ *r* / *2*
by *linarith*
also have ... < *r*
using *r* **by** *auto*
finally show |*measure* (*Ni n*) *M* −*measure N M*| < *r* **.**
qed
qed
next
define *bn* **where** *bn* ≡ (λ*n*. *LPm* (*Ni n*) *N*)
have *bn-nonneg*:⋀*n*. *bn n* ≥ *0*
by(*auto simp*: *bn-def*)
have *bn-tendsto*:(*bn* ⟶ *0*) *F*
using *h*(*2*) **by**(*auto simp*: *bn-def*)
fix *A*
assume *A*:*closedin mtopology A*
then have *A-meas*[*measurable*]:*A* ∈ *sets* (*borel-of mtopology*)
by(*simp add*: *borel-of-closed*)
show *Limsup F* (λ*x*. *measure* (*Ni x*) *A*) ≤ (*measure N A*)
proof(*cases A* = {})
assume *A-ne*:*A* ≠ {}
have *bdd*:*Limsup F* (λ*n*. *measure* (*Ni n*) *A*) ≤(*measure N* (⋃*a*∈*A*. *mball a* (*2* / *Suc m*))) + *1* / *Suc m* **for** *m*
proof −
have *Limsup F* (λ*n*. *measure* (*Ni n*) *A*)

≤ *Limsup F* (*λn. measure N* (⋃ *a*∈*A. mball a* (*bn n* + *1* / *Suc m*)) + *ereal* (*bn n* + *1* / *Suc m*))
by(*auto intro*!: *Limsup-mono eventually-mono*[*OF h*(*3*)] *LPm-less-then*(*1*)[*OF* - *h*(*1*)] *simp*: *bn-def*)
also have ... ≤ *Limsup F* (*λn. measure N* (⋃ *a*∈*A. mball a* (*bn n* + *1* / *Suc m*))) + *Limsup F* (*λn. bn n* + *1* / *Suc m*)
by(*rule ereal-Limsup-add-mono*)
also have ... = *Limsup F* (*λn. measure N* (⋃ *a*∈*A. mball a* (*bn n* + *1* / *Suc m*))) + *1* / *Suc m*
using *Limsup-add-ereal-right*[*OF F*,*of 1* / *Suc m bn*]
by(*simp add*: *lim-imp-Limsup*[*OF F tendsto-ereal*[*OF bn-tendsto*]])
also have ... ≤ *ereal* (*measure N* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))) + *1* / *Suc m*
proof −
have *Limsup F* (*λn. measure N* (⋃ *a*∈*A. mball a* (*bn n* + *1* / *Suc m*))) ≤ *measure N* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))
using *bn-nonneg*
by(*fastforce intro*!: *Limsup-bounded eventuallyI*[*THEN eventually-mp*[*OF* - *tendstoD*[*OF bn-tendsto*,*of 1* / *Suc m*]]]
N.finite-measure-mono)
thus *?thesis*
using *add-mono* **by** *blast*
qed
finally show *?thesis* **by** *simp*
qed
have *lim*:(*λm. ereal* ((*measure N* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))) + *1* / *Suc m*)) ⟶ *measure N A*
proof(*safe intro*!: *tendsto-ereal*[**where** *x*=*measure N A*] *tendsto-add*[**where** *b*=*0*,*simplified*])
show (*λm. measure N* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))) ⟶ *measure N A*
proof −
have *1*:(⋂ *m.* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))) = *A*
using *tendsto-mult*[*OF tendsto-const*[*of 2*] *LIMSEQ-Suc*[*OF lim-inverse-n'*]] *closedin-subset*[*OF A*]
by(*intro nbh-Inter-closure-of*[*OF A-ne*,*simplified closure-of-closedin*[*OF A*]] *decseq-SucI*)
(*auto simp*: *frac-le*)
have (*λm. measure N* (⋃ *a*∈*A. mball a* (*2* / *Suc m*))) ⟶ *measure N* (⋂ *m.* (⋃ *a*∈*A. mball a* (*2* / *Suc m*)))
by(*auto intro*!: *N.finite-Lim-measure-decseq nbh-decseq*[*OF decseq-SucI*] *simp*: *frac-le*)
thus *?thesis*
unfolding *1* **.**
qed
qed(*rule LIMSEQ-Suc*[*OF lim-inverse-n'*])
show *?thesis*
using *bdd* **by**(*auto intro*!: *Lim-bounded2*[*OF lim*])
qed(*simp add*: *Limsup-const*[*OF F*])
qed

next
  show $F = \bot \Longrightarrow$ *mweak-conv Ni N F*
    using *limitin-topspace*[*OF assms*(*1*)] **by**(*auto simp*: *inP-D mweak-conv-def*)
qed

lemma *mweak-conv-imp-converge*:
  assumes *separable-space mtopology*
    and *mweak-conv Ni N F*
  shows *limitin LPm.mtopology Ni N F*
proof −
  have *in-P*:$\forall_F$ *i in F*. *Ni i* $\in \mathcal{P}$ *N* $\in \mathcal{P}$
    using *limitin-topspace*[*OF assms*(*2*)]
    by(*fastforce intro*!: *eventually-mono*[*OF limitinD*[*OF assms*(*2*),
   of topspace (*weak-conv-topology mtopology*),*OF openin-topspace limitin-topspace*[*OF assms*(*2*)]]] *inP-I*)+
  consider $M = \{\}$ | $F = \bot$ | $M \neq \{\}$ $F \neq \bot$
    by *blast*
  thus *?thesis*
  proof *cases*
    case *1*
    then have *2*:*sets* (*borel-of mtopology*) = $\{\{\}\}$
      by (*metis space-borel-of space-empty-iff topspace-mtopology*)
    have $\forall_F$ *i in F*. *space* (*Ni i*) = *M space N* = *M*
      using *inP-D in-P*
     by(*auto intro*!: *eventually-mono*[*OF in-P*(*1*)] *cong*: *sets-eq-imp-space-eq simp*: *space-borel-of*)
    then have $\forall_F$ *i in F*. *Ni i* = *count-space* $\{\}$ *N* = *count-space* $\{\}$
      using *1* **by**(*auto simp*: *space-empty eventually-mono*)
    thus *?thesis*
      by(*auto intro*!: *limitin-eventually inP-I finite-measureI simp*: *2*)
  next
    show $F = \bot \Longrightarrow$ *limitin LPm.mtopology Ni N F*
      using *limitin-topspace*[*OF assms*(*2*)] **by**(*auto intro*!: *limitin-trivial inP-I*)
  next
    assume *M-ne*:$M \neq \{\}$ **and** *F*:$F \neq \bot$
    show *?thesis*
      unfolding *LPm.limitin-metric-dist-null dist-real-def tendsto-iff*
    proof *safe*
      interpret *mweak-conv-fin M d Ni N F*
        by(*auto intro*!: *inP-mweak-conv-fin in-P*)
      have *M*[*measurable*]: $M \in$ *sets N* $\forall_F$ *i in F*. $M \in$ *sets* (*Ni i*)
        by(*auto simp*: *sets-N borel-of-open eventually-mono*[*OF sets-Ni*])
      fix *r* :: *real*
      assume *r*[*arith*]: $0 < r$
      interpret *N*: *finite-measure N*
        using *in-P* **by**(*auto simp*: *inP-D*)
      define r' **where** $r' \equiv r / 5$
      have r'[*arith*]: $r' \leq r$ $0 < r'$
        by(*auto simp*: *r'-def*)

obtain *ai ri* **where** *airi*: $(\bigcup i.\ mball\ (ai\ i)\ (ri\ i)) = M$ $(\bigcup i.\ mcball\ (ai\ i)\ (ri\ i)) = M$
$\bigwedge i::nat.\ ai\ i \in M$ $\bigwedge i.\ 0 < ri\ i$ $\bigwedge i.\ ri\ i < r' / 2$
$\bigwedge i.\ measure\ N\ (mtopology\ frontier\text{-}of\ mball\ (ai\ i)\ (ri\ i)) = 0$
$\bigwedge i.\ measure\ N\ (mtopology\ frontier\text{-}of\ mcball\ (ai\ i)\ (ri\ i)) = 0$
using *frontier-measure-zero-balls*[*OF sets-N N.finite-measure-axioms M-ne half-gt-zero*[*OF r'(2)*] *assms(1)*]
by *blast*
have *meas*[*measurable*]: $\bigwedge a\ r.\ mball\ a\ r \in sets\ N$ $\forall_F\ j\ in\ F.\ \forall\ a\ r.\ mball\ a\ r \in sets\ (Ni\ j)$
$\bigwedge a\ r.\ mtopology\ frontier\text{-}of\ (mball\ a\ r) \in sets\ N$
$\forall_F\ j\ in\ F.\ \forall\ a\ r.\ mtopology\ frontier\text{-}of\ (mball\ a\ r) \in sets\ (Ni\ j)$
by(*auto simp*: *eventually-mono*[*OF sets-Ni*] *sets-N borel-of-open closedin-frontier-of borel-of-closed*)
have $\exists k.\ \forall l \ge k.\ |measure\ N\ (\bigcup i \in \{..l\}.\ mball\ (ai\ i)\ (ri\ i)) - measure\ N\ M| < r'$
proof −
have $(\lambda j.\ measure\ N\ (\bigcup i \in \{..j\}.\ mball\ (ai\ i)\ (ri\ i)))$
$\longrightarrow measure\ N\ (\bigcup\ (range\ (\lambda j.\ \bigcup i \in \{..j\}.\ mball\ (ai\ i)\ (ri\ i))))$
by(*rule N.finite-Lim-measure-incseq*) (*fastforce intro*!: *monoI*)+
hence $(\lambda j.\ measure\ N\ (\bigcup i \in \{..j\}.\ mball\ (ai\ i)\ (ri\ i))) \longrightarrow measure\ N\ M$
by (*metis UN-UN-flatten UN-atMost-UNIV airi(1)*)
thus *?thesis*
using *r'* **by**(*auto simp*: *LIMSEQ-def dist-real-def*)
qed
then obtain *k* **where** *k*: $measure\ N\ M - measure\ N\ (\bigcup i \in \{..k\}.\ mball\ (ai\ i)\ (ri\ i)) < r'$
using *space-N N.bounded-measure* **by** *fastforce*
define $\mathcal{A}$ **where** $\mathcal{A} = (\lambda J.\ \bigcup j \in J.\ mball\ (ai\ j)\ (ri\ j))$ ‘ $Pow\ \{..k\}$
have $\mathcal{A}$-*fin*: *finite* $\mathcal{A}$
by(*auto simp*: $\mathcal{A}$-*def*)
have $\mathcal{A}$-*ne*: $\mathcal{A} \neq \{\}$
by(*auto simp*: $\mathcal{A}$-*def*)
have $\forall_F\ n\ in\ F.\ |measure\ (Ni\ n)\ A - measure\ N\ A| < r'$ **if** $A \in \mathcal{A}$ **for** A
proof −
obtain *J* **where** *J*: $J \subseteq \{..k\}$ $A = (\bigcup j \in J.\ mball\ (ai\ j)\ (ri\ j))$
using ‹$A \in \mathcal{A}$› **by**(*auto simp*: $\mathcal{A}$-*def*)
hence *J-fin*: *finite J*
using *finite-nat-iff-bounded-le* **by** *blast*
have $measure\ N\ (mtopology\ frontier\text{-}of\ A) = measure\ N\ (mtopology\ frontier\text{-}of\ (\bigcup j \in J.\ mball\ (ai\ j)\ (ri\ j)))$
by(*auto simp*: *J*)
also have $... \le measure\ N\ (\bigcup\ ((frontier\text{-}of)\ mtopology$ ‘ $(\lambda j.\ mball\ (ai\ j)\ (ri\ j))$ ‘ $J))$
by(*rule N.finite-measure-mono*[*OF frontier-of-Union-subset*]) (*use J-fin* **in** *auto*)
also have $... \le (\sum j \in J.\ measure\ N\ (mtopology\ frontier\text{-}of\ mball\ (ai\ j)\ (ri\ j)))$
unfolding *image-image* **by**(*rule N.finite-measure-subadditive-finite*) (*use*

J-fin **in** *auto*)
also have ... = *0*
by(*simp add*: *airi*)
finally have *measure N* (*mtopology frontier-of A*) = *0*
by (*simp add*: *measure-le-0-iff*)
moreover have *A* ∈ *sets N*
by(*auto simp*: *J*(*2*))
ultimately show *?thesis*
using *mweak-conv-eq4 assms*(*2*) **by** (*fastforce simp*: *sets-N sets-Ni tendsto-iff dist-real-def*)
qed
hence *filter1*:∀$_F$ *n in F*. ∀ *A* ∈ 𝒜. |*measure* (*Ni n*) *A* − *measure N A*| < *r′*
by(*auto intro*!: 𝒜*-fin eventually-ball-finite*)
have *filter2*:∀$_F$ *n in F*. |*measure* (*Ni n*) *M* − *measure N M*| < *r′*
using *mweak-conv-imp-limit-space*[*OF assms*(*2*)] **by**(*auto simp*: *tendsto-iff dist-real-def*)
show ∀$_F$ *x in F*. |*LPm* (*Ni x*) *N* − *0*| < *r*
proof(*safe intro*!: *eventually-mono*[*OF eventually-conj*[*OF eventually-conj*[*OF finite-measure-Ni sets-Ni*] *eventually-conj*[*OF filter1 filter2*]]])
fix *n*
assume *n*:∀ *A*∈𝒜. |*measure* (*Ni n*) *A* − *measure N A*| < *r′* |*measure* (*Ni n*) *M* − *measure N M*| < *r′*
and *sets-Ni*[*measurable-cong*]: *sets* (*Ni n*) = *sets* (*borel-of mtopology*) **and** *finite-measure* (*Ni n*)
then have [*measurable*]:⋀*a r*. *mball a r* ∈ *sets* (*Ni n*)
⋀*a r*. *mtopology frontier-of mball a r* ∈ *sets* (*Ni n*) *M* ∈ *sets* (*Ni n*)
using *meas sets-N* **by** *auto*
have *space-Ni*: *space* (*Ni n*) = *M*
by(*simp add*: *sets-Ni space-borel-of cong*: *sets-eq-imp-space-eq*)
interpret *Ni*: *finite-measure Ni n* **by** *fact*
have *LPm* (*Ni n*) *N* < *r*
proof(*safe intro*!: *order.strict-trans1*[*OF LPm-imp-le*[*of 4* ∗ *r′*]])
fix *B*
assume *B* ∈ *sets* (*borel-of mtopology*)
hence [*measurable*]: *B* ∈ *sets N B* ∈ *sets* (*Ni n*)
by(*auto simp*: *sets-N*)
define *A* **where** *A* ≡ ⋃*j*∈{..*k*}∩{*j*. *mball* (*ai j*) (*ri j*) ∩ *B* ≠ {}}. *mball* (*ai j*) (*ri j*)
have *A-in*: *A* ∈ 𝒜
by(*auto simp*: 𝒜*-def A-def*)
have [*measurable*]: *A* ∈ *sets N A* ∈ *sets* (*Ni n*)
by(*auto simp*: *A-def*)
have *1*: *A* ⊆ (⋃ *a*∈*B*. *mball a r′*)
proof
fix *x*
assume *x* ∈ *A*
then obtain *j* **where** *j*:*j* ≤ *k mball* (*ai j*) (*ri j*) ∩ *B* ≠ {} *x* ∈ *mball* (*ai j*) (*ri j*)

by(*auto simp*: *A-def*)
then obtain *b* **where** *b*:*b* ∈ *mball* (*ai j*) (*ri j*) *b* ∈ *B*
by *blast*
have *d b x* ≤ *d b* (*ai j*) + *d* (*ai j*) *x*
using *b j* **by**(*auto intro*!: *triangle*)
also have ... < *r′* / *2* + *r′* / *2*
by(*rule add-strict-mono*, *insert b*(*1*) *airi*(*5*)[*of j*] *j*(*3*)) (*auto simp*: *commute*)
also have ... = *r′* **by** *auto*
finally show *x* ∈ (⋃ *a*∈*B*. *mball a r′*)
using *b*(*1*) *j*(*3*) **by**(*auto intro*!: *bexI*[**where** *x*=*b*] *b simp*: *mball-def*)
qed
have *2*: *B* ⊆ *A* ∪ (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)))
proof −
have *B* = *B* ∩ (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)) ∪ *B* ∩ (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)))
using *sets.sets-into-space*[*OF* ‹*B* ∈ *sets N*›] **by**(*auto simp*: *space-N*)
also have ... ⊆ *A* ∪ (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)))
by(*auto simp*: *A-def*)
finally show *?thesis* **.**
qed
have *3*: *measure N* (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*))) < *r′*
using *N.finite-measure-compl k space-N* **by** *auto*
have *4*: *measure* (*Ni n*) (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*))) < *3* ∗ *r′*
proof −
have *measure* (*Ni n*) (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)))
= *measure* (*Ni n*) *M* − *measure* (*Ni n*) (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*))
using *Ni.finite-measure-compl space-Ni* **by** *auto*
also have ... < *measure N M* + *r′*− (*measure N* (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)) − *r′*)
by(*rule diff-strict-mono*,*insert n*) (*auto simp*: *abs-diff-less-iff* 𝒜*-def*)
also have ... = *measure N* (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*))) + *2* ∗ *r′*
using *N.finite-measure-compl diff-add-cancel space-N* **by** *auto*
finally show *?thesis*
using *3* **by** *auto*
qed
show *measure* (*Ni n*) *B* ≤ *measure N* (⋃ *a*∈*B*. *mball a* (*4* ∗ *r′*)) + *4* ∗ *r′*
proof −
have *measure* (*Ni n*) *B* ≤ *measure* (*Ni n*) (*A* ∪ (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*))))
by(*auto intro*!: *Ni.finite-measure-mono*[*OF 2*])
also have ... ≤ *measure* (*Ni n*) *A* + *measure* (*Ni n*) (*M* − (⋃ *j*≤*k*. *mball* (*ai j*) (*ri j*)))
by(*auto intro*!: *Ni.finite-measure-subadditive*)
also have ... < *measure N A* + *4* ∗ *r′*
using *4 A-in n* **by**(*auto simp*: *abs-diff-less-iff*)
also have ... ≤ *measure N* (⋃ *a*∈*B*. *mball a r′*) + *4* ∗ *r′*
by(*auto intro*!: *N.finite-measure-mono*[*OF 1*] *borel-of-open simp*: *sets-N*)
also have ... ≤ *measure N* (⋃ *a*∈*B*. *mball a* (*4* ∗ *r′*)) + *4* ∗ *r′*

using *mball-subset-concentric*[*of* r' *4* $*$ r']
by(*auto intro*!: *N.finite-measure-mono borel-of-open simp*: *sets-N*)
finally show *?thesis* **by** *simp*
qed
show *measure N B* $\leq$ *measure* (*Ni n*) ($\bigcup$ *a*$\in$*B*. *mball a* (*4* $*$ r')) + *4* $*$ r'
proof −
have *measure N B* $\leq$ *measure N* (*A* $\cup$ (*M* − ($\bigcup$ *j*$\leq$*k*. *mball* (*ai j*) (*ri j*))))
by(*auto intro*!: *N.finite-measure-mono*[*OF 2*])
also have ... $\leq$ *measure N A* + *measure N* (*M* − ($\bigcup$ *j*$\leq$*k*. *mball* (*ai j*) (*ri j*)))
by(*auto intro*!: *N.finite-measure-subadditive*)
also have ... < *measure* (*Ni n*) *A* + *2* $*$ r'
using *3 A-in n* **by**(*auto simp*: *abs-diff-less-iff*)
also have ... $\leq$ *measure* (*Ni n*) ($\bigcup$ *a*$\in$*B*. *mball a* r') + *2* $*$ r'
by(*auto intro*!: *Ni.finite-measure-mono*[*OF 1*] *borel-of-open simp*: *sets-N*)
also have ... $\leq$ *measure* (*Ni n*) ($\bigcup$ *a*$\in$*B*. *mball a* (*4* $*$ r')) + *2* $*$ r'
using *mball-subset-concentric*[*of* r' *4* $*$ r']
by(*auto intro*!: *Ni.finite-measure-mono borel-of-open simp*: *sets-N*)
finally show *?thesis* **by** *simp*
qed
qed (*auto simp*: r'*-def*)
thus |*LPm* (*Ni n*) *N* − *0*| < *r*
by *simp*
qed
qed (*use in-P* **in** *auto*)
qed
qed

corollary *conv-iff-mweak-conv*: *separable-space mtopology* $\Longrightarrow$ *limitin LPm.mtopology Ni N F* $\longleftrightarrow$ *mweak-conv Ni N F*
using *converge-imp-mweak-conv mweak-conv-imp-converge* **by** *blast*

4.3 Separability

lemma *LPm-countable-base*:
assumes *ai*:*mdense* (*range ai*)
shows *LPm.mdense*
((λ(*k*,*bi*). *sum-measure*
(*borel-of mtopology*) {..*k*}
(λ*i*. *scale-measure* (*ennreal* (*bi i*)) (*return* (*borel-of mtopology*) (*ai i*))))
‘ (*SIGMA k*:(*UNIV* :: *nat set*). ({..*k*} $\rightarrow_E$ $\mathbb{Q}$ $\cap$ {*0*..}))) (**is** *LPm.mdense ?D*)
proof −
have *sep*:*separable-space mtopology*
using *ai* **by**(*auto simp*: *separable-space-def2 intro*!: *exI*[**where** *x*=*range ai*])
have *ai-in*: $\bigwedge$*i*. *ai i* $\in$ *M*
by (*meson ai mdense-def2 range-subsetD*)

hence *M-ne*:*M* ≠ {}
by *blast*
show *?thesis*
unfolding *LPm.mdense-def3*
proof
show *goal1*:*?D* ⊆ 𝒫
proof *safe*
fix *bi* :: *nat* ⇒ *real* **and** *k* :: *nat*
assume *h*: *bi* ∈ {..*k*} →$_E$ ℚ ∩ {*0*..}
show *sum-measure* (*borel-of mtopology*) {..*k*}
(λ*i*. *scale-measure* (*ennreal* (*bi i*)) (*return* (*borel-of mtopology*) (*ai i*))) ∈ 𝒫
by(*auto simp*: 𝒫*-def emeasure-sum-measure intro*!: *finite-measureI*)
qed
show ∀ *x*∈𝒫. ∃ *xn*. *range xn* ⊆ *?D* ∧ *limitin LPm.mtopology xn x sequentially*
proof
fix *N*
assume *N* ∈ 𝒫
then have *sets-N*[*measurable-cong*]: *sets N* = *sets* (*borel-of mtopology*)
and *space-N*:*space N* = *M* **and** *finite-measure N*
by(*auto simp*: 𝒫*-def space-borel-of cong*: *sets-eq-imp-space-eq*)
then interpret *N*: *finite-measure N* **by** *simp*
have [*measurable*]:⋀*a r*. *mball a r* ∈ *sets N*
by(*auto simp*: *sets-N borel-of-open*)
have *ai-in*′[*measurable*]: ⋀*i*. *ai i* ∈ *space N*
by(*auto simp*: *ai-in space-N*)
have (λ*i*. *measure N* (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*))) ⟶ *measure N* (*space N*) **for** *m*
proof −
have *1*:(⋃ *i*. (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*))) = *space N*
using *mdense-balls-cover*[*OF ai*,*of 1* / *Suc m*] **by**(*auto simp*: *space-N*)
have (λ*i*. *measure N* (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*)))
⟶ *measure N* (⋃ *i*. (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*)))
by(*rule N*.*finite-Lim-measure-incseq*) (*fastforce intro*!: *monoI*)+
thus *?thesis*
unfolding *1* **.**
qed
hence ∃ *k*. ∀ *i*≥*k*. |*measure N* (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*)) − *measure N* (*space N*)| < *1* / *Suc m* **for** *m*
unfolding *LIMSEQ-def dist-real-def* **by** *fastforce*
then obtain *k* **where**
⋀*i m*. *i* ≥ *k m* ⟹ |*measure N* (⋃*j*≤*i*. *mball* (*ai j*) (*1* / *Suc m*)) − *measure N* (*space N*)| < *1* / *Suc m*
by *metis*
hence *k*: ⋀*m*. *measure N* (*space N*) − *measure N* (⋃*j*≤*k m*. *mball* (*ai j*) (*1* / *Suc m*)) < *1* / *Suc m*
using *N*.*bounded-measure* **by** *auto*
define *Ami*
where *Ami* ≡ (λ*m i*. (⋃*j*<*Suc i*. *mball* (*ai j*) (*1* / *Suc m*)) − (⋃*j*<*i*. *mball*

(*ai j*) (*1* / *Suc m*)))
have *Ami-disj*: ⋀*m*. *disjoint-family* (*Ami m*)
by(*fastforce simp*: *Ami-def intro*!: *disjoint-family-Suc*)
have *Ami-def'*: *Ami* = (λ*m i*. *mball* (*ai i*) (*1* / *Suc m*) − (⋃*j*<*i*. *mball* (*ai j*) (*1* / *Suc m*)))
by (*standard*, *standard*) (*auto simp*: *Ami-def less-Suc-eq*)
have *Ami-subs*: *Ami m i* ⊆ *mball* (*ai i*) (*1* / *Suc m*) **for** *m i*
by(*auto simp*: *Ami-def'*)
have *Ami-un*: (⋃*i*≤*j*. *Ami m i*) = (⋃*i*≤*j*. *mball* (*ai i*) (*1* / *Suc m*)) **for** *m j*
proof
show (⋃*i*≤*j*. *mball* (*ai i*) (*1* / *real* (*Suc m*))) ⊆ (⋃*i*≤*j*. *Ami m i*)
proof(*induction j*)
case *0*
then show *?case*
by(*auto simp*: *Ami-def*)
next
case *ih*:(*Suc j*)
have (⋃*i*≤*Suc j*. *mball* (*ai i*) (*1* / *real* (*Suc m*)))
= (⋃*i*≤*j*. *mball* (*ai i*) (*1* / (*Suc m*))) ∪ *mball* (*ai* (*Suc j*)) (*1* / *Suc m*)
by(*fastforce simp*: *le-Suc-eq*)
also have ... = (⋃*i*≤*j*. *mball* (*ai i*) (*1* / (*Suc m*))) ∪
(*mball* (*ai* (*Suc j*)) (*1* / *Suc m*) − (⋃*i*<*Suc j*. *mball* (*ai i*) (*1* / (*Suc m*))))
by *fastforce*
also have ... ⊆ (⋃*i*≤*Suc j*. *Ami m i*)
proof −
have (*mball* (*ai* (*Suc j*)) (*1* / *Suc m*) − (⋃*i*<*Suc j*. *mball* (*ai i*) (*1* / (*Suc m*))))
⊆ (⋃*i*≤*Suc j*. *Ami m i*)
using *Ami-def'* **by** *blast*
thus *?thesis*
using *ih* **by**(*fastforce simp*: *le-Suc-eq*)
qed
finally show *?case* **.**
qed
qed(*use Ami-subs* **in** *auto*)
have *sets-Ami*[*measurable*]: ⋀*m i*. *Ami m i* ∈ *sets N*
by(*auto simp*: *Ami-def*)
have ∃ *qmi*. *qmi* ∈({..*k m*} $\rightarrow_E$ ℚ ∩ {*0*..}) ∧ (∑ *i*≤*k m*. |*measure N* (*Ami m i*) − *qmi i*|) < *1* / *Suc m* **for** *m*
proof −
have ∃ *qmi*∈ ℚ ∩ {*0*..}. *measure N* (*Ami m i*) − *qmi* < *1* / (*real* (*Suc m*) ∗ *real* (*Suc* (*k m*))) ∧
qmi ≤ *measure N* (*Ami m i*) **if** *i* ≤ *k m* **for** *i*
proof(*cases measure N* (*Ami m i*) = *0*)
case *True*
then show *?thesis*
by(*auto intro*!: *bexI*[**where** *x*=*0*])

next
case *False*
hence $max\ 0\ (measure\ N\ (Ami\ m\ i) - 1\ /\ (real\ (Suc\ m) * real\ (Suc\ (k\ m)))) < measure\ N\ (Ami\ m\ i)$
by(*auto simp*: *zero-less-measure-iff*)
from *of-rat-dense*[*OF this*] **obtain** *q* **where**
$q{:}0 < real\text{-}of\text{-}rat\ q$ $measure\ N\ (Ami\ m\ i) - 1\ /\ (real\ (Suc\ m) * real\ (Suc\ (k\ m))) < real\text{-}of\text{-}rat\ q$
$real\text{-}of\text{-}rat\ q < measure\ N\ (Ami\ m\ i)$
by *auto*
hence $real\text{-}of\text{-}rat\ q \in \mathbb{Q} \cap \{0..\}$
by *auto*
with *q*(*2*,*3*) **show** *?thesis*
by(*auto intro*!: *bexI*[**where** *x*=*real-of-rat q*])
qed
then obtain *qmi* **where** $qmi{:}\bigwedge i.\ i \le k\ m \Longrightarrow qmi\ i \in \mathbb{Q} \cap \{0..\}$
$\bigwedge i.\ i \le k\ m \Longrightarrow measure\ N\ (Ami\ m\ i) - qmi\ i < 1\ /\ (real\ (Suc\ m) * real\ (Suc\ (k\ m)))$
$\bigwedge i.\ i \le k\ m \Longrightarrow qmi\ i \le measure\ N\ (Ami\ m\ i)$
by *metis*
have *2*: $(\sum i{\le}k\ m.\ |measure\ N\ (Ami\ m\ i) - qmi\ i|) < 1\ /\ Suc\ m$
proof −
have $\bigwedge i.\ i \le k\ m \Longrightarrow |measure\ N\ (Ami\ m\ i) - qmi\ i| < 1\ /\ (real\ (Suc\ m) * real\ (Suc\ (k\ m)))$
using *qmi* **by** *auto*
hence $(\sum i{\le}k\ m.\ |measure\ N\ (Ami\ m\ i) - qmi\ i|) < (\sum i{\le}k\ m.\ 1\ /\ (real\ (Suc\ m) * real\ (Suc\ (k\ m))))$
by(*intro sum-strict-mono*) *auto*
also have ... = *1* / *Suc m*
by *auto*
finally show *?thesis* **.**
qed
show *?thesis*
using *qmi 2* **by**(*intro exI*[**where** $x{=}\lambda i{\in}\{..k\ m\}.\ qmi\ i$]) *force*
qed

hence $\exists\, qmi.\ \forall\, m.\ qmi\ m \in (\{..k\ m\} \rightarrow_E \mathbb{Q} \cap \{0..\}) \wedge (\sum i{\le}k\ m.\ |measure\ N\ (Ami\ m\ i) - qmi\ m\ i|) < 1\ /\ Suc\ m$
by(*intro choice*) *auto*
then obtain *qmi* **where** $qmi{:}\ \bigwedge m.\ qmi\ m \in (\{..k\ m\} \rightarrow_E \mathbb{Q} \cap \{0..\})$
$\bigwedge m.\ (\sum i{\le}k\ m.\ |measure\ N\ (Ami\ m\ i) - qmi\ m\ i|) < 1\ /\ Suc\ m$
by *blast*
define *Ni* **where** $Ni \equiv (\lambda i.\ sum\text{-}measure\ N\ \{..k\ i\}\ (\lambda j.\ scale\text{-}measure\ (qmi\ i\ j)\ (return\ N\ (ai\ j))))$
have *NiD*:$Ni\ i \in\ ?D$ **for** *i*
using *qmi* **by**(*auto simp*: *Ni-def image-def intro*!: *exI*[**where** *x*=*k i*] *bexI*[**where** *x*=*qmi i*]
cong: *return-cong*[*OF sets-N*] *sum-measure-cong*[*OF sets-N refl*])

with *goal1* **have** *NiP*: $\bigwedge i.$ *Ni i* ∈ $\mathcal{P}$ **by** *auto*
hence *Nifin*: $\bigwedge i.$ *finite-measure* (*Ni i*)
and *sets-Ni'*[*measurable-cong*]: $\bigwedge i.$ *sets* (*Ni i*) = *borel-of mtopology*
by(*auto simp*: *inP-D*)
interpret *mweak-conv-fin M d Ni N sequentially*
using *NiP* $\mathcal{P}$*-def* ‹*N* ∈ $\mathcal{P}$› *inP-mweak-conv-fin-all* **by** *blast*
show ∃ *xn. range xn* ⊆ *?D* ∧ *limitin LPm.mtopology xn N sequentially*
proof(*safe intro*!: *exI*[**where** *x*=*Ni*] *mweak-conv-imp-converge sep*)
show *mweak-conv-seq Ni N*
unfolding *mweak-conv-eq1 LIMSEQ-def*
proof *safe*
fix *g* :: *'a* ⇒ *real* **and** *K r* :: *real*
assume *h*: *uniformly-continuous-map Self euclidean-metric g* ∀ *x*∈*M*. |*g x*| ≤ *K* **and** *r*[*arith*]: *r* > *0*
have [*measurable*]:*g* ∈ *borel-measurable N*
using *continuous-map-measurable*[*OF uniformly-continuous-imp-continuous-map*[*OF h*(*1*)]]
by(*auto simp*: *borel-of-euclidean mtopology-of-def cong*: *measurable-cong-sets sets-N*)
have *gK*: $\bigwedge x.$ *x* ∈ *space N* ⟹ |*g x*| ≤ *K*
using *h*(*2*) **by**(*auto simp*: *space-N*)
have *K-nonneg*: *K* ≥ *0*
using *h*(*2*) *M-ne* **by** *auto*
have ∃ *m*. *2* ∗ *K* / *Suc m* < *r* / *2*
proof (*cases K* = *0*)
assume *K*:*K* ≠ *0*
then have *r* / *2* ∗ (*1* / (*2* ∗ *K*)) > *0*
using *K-nonneg* **by** *auto*
then obtain *m* **where** *1* / *Suc m* < *r* / *2* ∗ (*1* / (*2* ∗ *K*))
by (*meson nat-approx-posE*)
from *mult-strict-right-mono*[*OF this*,*of 2* ∗ *K*] **show** *?thesis*
using *K K-nonneg* **by** *auto*
qed *simp*
then obtain *m1* **where** *m1*: *2* ∗ *K* / *Suc m1* < *r* / *2* **by** *auto*
obtain δ **where** δ: δ > *0*
$\bigwedge x\, y.$ *x* ∈ *M* ⟹ *y* ∈ *M* ⟹ *d x y* < δ ⟹ |*g x* − *g y*| < *r* / *2* ∗ (*1* / (*1* + *measure N* (*space N*)))
using *conjunct2*[*OF h*(*1*)[*simplified uniformly-continuous-map-def*], *rule-format*,*of* (*r* / *2*) ∗ (*1* / (*1* + *measure N* (*space N*)))] *measure-nonneg*[*of N space N*] *r*
unfolding *mdist-Self mspace-Self mdist-euclidean-metric dist-real-def* **by** *auto*
obtain *m2* **where** *m2*: *1* / *Suc m2* < δ
using δ(*1*) *nat-approx-posE* **by** *blast*
define *m* **where** *m* ≡ *max m1 m2*
then have *m*:*1* / *Suc m* ≤ *1* / *real* (*Suc m1*) *1* / *Suc m* ≤ *1* / *real* (*Suc m2*)
by (*simp-all add*: *frac-le*)
show ∃ *no*. ∀ *n*≥*no*. *dist* (∫ *x*. *g x* ∂*Ni n*) (∫ *x*. *g x* ∂*N*) < *r*

unfolding *dist-real-def*
proof(*safe intro*!: *exI*[**where** *x*=*m*])
fix *n*
assume $n \geq m$
then have *n*:*1* / *Suc n* ≤ *1* / *real* (*Suc m*)
by (*simp add*: *frac-le*)
have *int1*[*measurable*]: *integrable* (*return N* (*ai j*)) *g* **for** *j*
unfolding *integrable-iff-bounded*
proof *safe*
show ($\int^+$ *x*. *ennreal* (*norm* (*g x*)) ∂*return N* (*ai j*)) < ∞
by(*rule order.strict-trans1*[*OF nn-integral-mono*[**where** *v*=λ*x*. *ennreal*
K]])
(*auto simp*: *ai-in′ gK intro*!: *ennreal-leI*)
qed *simp*
have *int2*[*measurable*]: ⋀*A*. *A* ∈ *sets N* ⟹ *integrable N* (*indicat-real A*)
using *N*.*fmeasurable-eq-sets fmeasurable-def* **by** *blast*
have *intg*: *integrable N g*
by(*auto intro*!: *N*.*integrable-const-bound*[**where** *B*=*K*] *gK*)
show |(∫ *x*. *g x* ∂*Ni n*) − (∫ *x*. *g x* ∂*N*)| < *r* (**is** *?lhs* < -)
proof −
have *?lhs* = |(∑ *i*≤*k n*. ∫ *x*. *g x* ∂*scale-measure* (*qmi n i*) (*return N*
(*ai i*))) − (∫ *x*. *g x* ∂*N*)|
by(*simp add*: *Ni-def integral-sum-measure*[*OF* - *integrable-scale-measure*[*OF*
int1]])
also have ... = |(∑ *i*≤*k n*. *qmi n i* ∗ *g* (*ai i*)) − (∫ *x*. *g x* ∂*N*)|
proof −
{
fix *i*
assume *i*:*i* ≤ *k n*
then have (∫ *x*. *g x* ∂*scale-measure* (*qmi n i*) (*return N* (*ai i*))) =
qmi n i ∗ *g* (*ai i*)
using *integral-scale-measure*[*OF* - *int1*,*of qmi n i*] *qmi*(*1*)[*of n*]
int1
by(*fastforce simp*: *integral-return ai-in′*)
}
thus *?thesis*
by *simp*
qed
also have ... = |(∑ *i*≤*k n*. *qmi n i* ∗ *g* (*ai i*)) − (∑ *i*≤*k n*. *measure N*
(*Ami n i*) ∗ *g* (*ai i*))
+ ((∑ *i*≤*k n*. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g*
x ∂*N*))|
by *simp*
also have ... ≤ |(∑ *i*≤*k n*. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∑ *i*≤*k*
n. *qmi n i* ∗ *g* (*ai i*))|
+ |(∑ *i*≤*k n*. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g*
x ∂*N*)|
by *auto*
also have ... = |∑ *i*≤*k n*. (*measure N* (*Ami n i*) − *qmi n i*) ∗ *g* (*ai i*)|

+ |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g* *x* ∂*N*)|
by(*simp add*: *sum-subtractf left-diff-distrib*)
also have ... ≤ (∑ i≤k n. |(*measure N* (*Ami n i*) − *qmi n i*) ∗ *g* (*ai* *i*)|)
+ |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g* *x* ∂*N*)|
by *simp*
also have ... = (∑ i≤k n. |*measure N* (*Ami n i*) − *qmi n i*| ∗ |*g* (*ai i*)|)
+ |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g* *x* ∂*N*)|
by (*simp add*: *abs-mult*)
also have ... ≤ (∑ i≤k n. |*measure N* (*Ami n i*) − *qmi n i*| ∗ *K*)
+ |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g* *x* ∂*N*)|
by(*auto intro*!: *sum-mono mult-left-mono gK*[*OF ai-in'*])
also have ... = (∑ i≤k n. |*measure N* (*Ami n i*) − *qmi n i*|) ∗ *K*
+ |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g* *x* ∂*N*)|
by (*simp add*: *sum-distrib-right*)
also have ... ≤ *1* / *Suc n* ∗ *K* + |(∑ i≤k n. *measure N* (*Ami n i*) ∗ *g* (*ai i*)) − (∫ *x*. *g x* ∂*N*)|
proof −
have (∑ i≤k n. |*measure N* (*Ami n i*) − *qmi n i*|) ∗ *K* ≤ *1* / *Suc n* ∗ *K*
by(*rule mult-right-mono*) (*use qmi*(*2*)[*of n*] *K-nonneg* **in** *auto*)
thus *?thesis* **by** *simp*
qed
also have ... = *K* / *Suc n* + |(∑ i≤k n. (∫ *x*. *indicator* (*Ami n i*) *x* ∗ *g* (*ai i*) ∂*N*)) − (∫ *x*. *g x* ∂*N*)|
by *auto*
also have ... = *K* / *Suc n* + |(∫ *x*. (∑ i≤k n. *indicator* (*Ami n i*) *x* ∗ *g* (*ai i*)) ∂*N*) − (∫ *x*. *g x* ∂*N*)|
proof −
have (∑ i≤k n. (∫ *x*. *indicator* (*Ami n i*) *x* ∗ *g* (*ai i*) ∂*N*))
= (∫ *x*. (∑ i≤k n. *indicator* (*Ami n i*) *x* ∗ *g* (*ai i*)) ∂*N*)
by(*rule integral-sum'*[*symmetric*]) (*use int2* **in** *auto*)
thus *?thesis*
by *simp*
qed
also have ... = *K* / *Suc n*
+ |(∫ *x*. (∑ i≤k n. *indicat-real* (*Ami n i*) *x* ∗ *g* (*ai i*)) ∂*N*)
− ((∫ *x*. (∑ i≤k n. *indicat-real* (*Ami n i*) *x* ∗ *g x*) ∂*N*)
+ (∫ *x*. *indicat-real* (*space N* − (⋃ i≤k n. *Ami n i*)) *x* ∗*g x* ∂*N*))|
proof −
have ∗:*indicat-real* (⋃ i≤k n. *Ami n i*) *x* = (∑ i≤k n. *indicat-real* (*Ami n i*) *x*) **for** *x*
by(*auto intro*!: *indicator-UN-disjoint Ami-disj disjoint-family-on-mono*[*OF*

- *Ami-disj*[*of n*]])
hence ($\int x.$ ($\sum i{\le}k$ *n. indicat-real* (*Ami n i*) $x * g\ x$) ∂N)
+ ($\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x *g\ x\ \partial N$)
= ($\int x.$ *indicat-real* ($\bigcup i{\le}k$ *n. Ami n i*) $x * g\ x\ \partial N$)
+ ($\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x *g\ x\ \partial N$)
by (*simp add*: *sum-distrib-right*)
also have ... = ($\int x.$ *indicat-real* ($\bigcup i{\le}k$ *n. Ami n i*) $x * g\ x$
+ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x * g\ x$
∂N)
by(*rule Bochner-Integration.integral-add*[*symmetric*])
(*auto intro*!: *integrable-mult-indicator*[**where** *'b=real*,*simplified*]
intg)
also have ... = ($\int x.\ g\ x\ \partial N$)
by(*auto intro*!: *Bochner-Integration.integral-cong*) (*auto simp*:
indicator-def)
finally show *?thesis* **by** *simp*
qed
also have ... = *K* / *Suc n*
+ |($\sum i{\le}k$ *n.* $\int x.$ *indicat-real* (*Ami n i*) $x * g$ (*ai i*) ∂N)
− (($\sum i{\le}k$ *n.* $\int x.$ *indicat-real* (*Ami n i*) $x * g\ x\ \partial N$)
+ ($\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) x
$*g\ x\ \partial N$))|
proof −
have ∗: ($\int x.$ ($\sum i{\le}k$ *n. indicat-real* (*Ami n i*) $x * g$ (*ai i*)) ∂N)
= ($\sum i{\le}k$ *n.* $\int x.$ *indicator* (*Ami n i*) $x * g$ (*ai i*) ∂N)
by(*rule Bochner-Integration.integral-sum*) (*use int2* **in** *auto*)
have ∗∗: ($\int x.$ ($\sum i{\le}k$ *n. indicat-real* (*Ami n i*) $x * g\ x$) ∂N)
= ($\sum i{\le}k$ *n.* $\int x.$ *indicat-real* (*Ami n i*) $x * g\ x\ \partial N$)
by(*rule Bochner-Integration.integral-sum*)
(*auto intro*!: *integrable-mult-indicator*[**where** *'b=real*,*simplified*]
intg)
show *?thesis*
unfolding ∗ ∗∗ **by** *simp*
qed
also have ... = *K* / *Suc n*
+ |($\sum i{\le}k$ *n.* ($\int x.$ *indicat-real* (*Ami n i*) $x * g$ (*ai i*) ∂N) − ($\int x.$
indicat-real (*Ami n i*) $x * g\ x\ \partial N$))
− ($\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x * g\ x\ \partial N$)|
by(*simp add*: *sum-subtractf*)
also have ... ≤ *K* / *Suc n*
+ |($\sum i{\le}k$ *n.* ($\int x.$ *indicat-real* (*Ami n i*) $x * g$ (*ai i*) ∂N) − ($\int x.$
indicat-real (*Ami n i*) $x * g\ x\ \partial N$))|
+ |$\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x * g\ x\ \partial N$|
by *linarith*
also have ... ≤ *K* / *Suc n*
+ |$\sum i{\le}k$ *n.* ($\int x.$ *indicat-real* (*Ami n i*) $x *$ (*g* (*ai i*) − *g*
x) ∂N)|
+ |$\int x.$ *indicat-real* (*space N* − ($\bigcup i{\le}k$ *n. Ami n i*)) $x * g$
$x\ \partial N$|

proof −
have ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ *g* (*ai i*) ∂*N*) − ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ *g x* ∂*N*)) = ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ *g* (*ai i*) − *indicat-real* (*Ami n i*) *x* ∗ *g x* ∂*N*))
by(*rule Finite-Cartesian-Product.sum-cong-aux*[*OF Bochner-Integration.integral-diff*[*symmetric*]]) (*auto intro*!: *integrable-mult-indicator*[**where** ′*b*=*real*,*simplified*] *intg int2*)
also have ... = ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ (*g* (*ai i*) − *g x*) ∂*N*))
by(*simp add*: *right-diff-distrib*)
finally show *?thesis* **by** *simp*
qed
also have ... ≤ *1* / *Suc n* ∗ *K* + *r* / *2* + *1* / *Suc n* ∗ *K*
proof −
have ∗:|$\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ (*g* (*ai i*) − *g x*) ∂*N*)| ≤ *r* / *2*
proof −
have |$\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ (*g* (*ai i*) − *g x*) ∂*N*)|
≤ ($\sum i \le k\ n.$ |$\int x.$ *indicat-real* (*Ami n i*) *x* ∗ (*g* (*ai i*) − *g x*) ∂*N*|)
by(*rule sum-abs*)
also have ... ≤ ($\sum i \le k\ n.$ ($\int x.$ |*indicat-real* (*Ami n i*) *x* ∗ (*g* (*ai i*) − *g x*)| ∂*N*))
by(*auto intro*!: *sum-mono*)
also have ... = ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ |(*g* (*ai i*) − *g x*)| ∂*N*))
by(*auto intro*!: *Finite-Cartesian-Product.sum-cong-aux Bochner-Integration.integral-cong*
simp: *abs-mult*)
also have ... ≤ ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x* ∗ ($\bigsqcup$ *y*∈*Ami n i*. |*g* (*ai i*) − *g y*|) ∂*N*))
proof(*rule sum-mono*[*OF integral-mono*])
fix *i x*
show *indicat-real* (*Ami n i*) *x* ∗ |*g* (*ai i*) − *g x*|
≤ *indicat-real* (*Ami n i*) *x* ∗ ($\bigsqcup$ *y*∈*Ami n i*. |*g* (*ai i*) − *g y*|)
using *gK gK*[*OF ai-in*′[*of i*]] *sets.sets-into-space*[*OF sets-Ami*[*of n i*]]
by(*fastforce simp*: *indicator-def intro*!: *cSUP-upper bdd-aboveI*[**where** *M*=*2* ∗ *K*])
qed(*auto intro*!: *integrable-mult-indicator*[**where** ′*b*=*real*,*simplified*] *intg int2*)
also have ... ≤ ($\sum i \le k\ n.$ ($\int x.$ *indicat-real* (*Ami n i*) *x*
∗ (*r* / *2* ∗ (*1* / (*1* + *measure N* (*space N*)))) ∂*N*))
proof(*rule sum-mono*[*OF integral-mono*])
fix *i x*
show *indicat-real* (*Ami n i*) *x* ∗ ($\bigsqcup$ *y*∈*Ami n i*. |*g* (*ai i*) − *g y*|)
≤ *indicat-real* (*Ami n i*) *x* ∗ (*r* / *2* ∗ (*1* / (*1* + *measure N* (*space N*))))
proof −
{

```
                  assume x:x ∈ Ami n i
                    have (⨆ y∈Ami n i. |g (ai i) − g y|) ≤ r / 2 ∗ (1 / (1 +
measure N (space N)))
                  proof(safe intro!: cSup-le-iff [THEN iffD2])
                    fix y
                    assume y:y ∈ Ami n i
                  with Ami-subs[of n i] have y ∈ mball (ai i) (1 / real (Suc n))
                      by auto
                    with δ(2) n m m2
                   show |g (ai i) − g y| ≤ r / 2 ∗ (1 / (1 + measure N (space
N)))
                      by fastforce
                     qed(insert x gK gK[OF ai-in′[of i]] sets.sets-into-space[OF
sets-Ami[of n i]],
                      fastforce intro!: bdd-aboveI[where M=2∗K])+
                }
                thus ?thesis
                  by(auto simp: indicator-def)
              qed
            qed(auto intro!: integrable-mult-indicator[where ′b=real,simplified]
intg int2)
            also have ... ≤ (∑ i≤k n. measure N (Ami n i)) ∗ (r / 2 ∗ (1 / (1
+ measure N (space N))))
              by (simp only: sum-distrib-right) auto
            also have ... = measure N (⋃ i≤k n. (Ami n i)) ∗ (r / 2 ∗ (1 / (1
+ measure N (space N))))
                by(auto intro!: N.finite-measure-finite-Union[symmetric] dis-
joint-family-on-mono[OF - Ami-disj[of n]])
           also have ... ≤ (r / 2) ∗ (measure N (space N) ∗ (1 / (1 + measure
N (space N))))
              using r measure-nonneg N.bounded-measure
               by(auto simp del: times-divide-eq-left times-divide-eq-right intro!:
mult-right-mono)
            also have ... ≤ r / 2
          by(intro mult-left-le) (auto simp: divide-le-eq-1 intro!: add-pos-nonneg)
            finally show ?thesis .
          qed
           have ∗∗: |∫ x. indicat-real (space N − (⋃ i≤k n. Ami n i)) x ∗ g x
∂N| ≤ 1 / Suc n ∗ K
          proof −
           have |∫ x. indicat-real (space N − (⋃ i≤k n. Ami n i)) x ∗ g x ∂N|
                 ≤ (∫ x. |indicat-real (space N − (⋃ i≤k n. Ami n i)) x ∗ g x|
∂N)
              by simp
           also have ... = (∫ x. indicat-real (space N − (⋃ i≤k n. Ami n i)) x
∗ |g x| ∂N)
              by(auto intro!: Bochner-Integration.integral-cong simp: abs-mult)
           also have ... ≤ (∫ x. indicat-real (space N − (⋃ i≤k n. Ami n i)) x
∗ K ∂N)
```

```
                    by(rule integral-mono,insert gK)
                      (auto intro!: integrable-mult-indicator[where 'b=real,simplified]
intg int2
                            simp: ordered-semiring-class.mult-left-mono)
                  also have ... = measure N (space N − (⋃i≤k n. Ami n i)) ∗ K
                    by simp
                 also have ... = (measure N (space N) − measure N (⋃j≤k n. mball
(ai j) (1 / real (Suc n)))) ∗ K
                    unfolding Ami-un by(simp add: N.finite-measure-compl)
                  also have ... ≤ 1 / Suc n ∗ K
                          by (metis k[of n] K-nonneg less-eq-real-def mult.commute
mult-left-mono)
                  finally show ?thesis .
                qed
                show ?thesis
                  using ∗ ∗∗ by auto
              qed
              also have ... =  2 ∗ K / Suc n + r / 2
                by simp
              also have ... ≤ 2 ∗ K / Suc m + r / 2
                using K-nonneg by (simp add: ‹m ≤ n› frac-le)
              also have ... ≤ 2 ∗ K / Suc m1 + r / 2
                using K-nonneg divide-inverse m(1) mult-left-mono by fastforce
              also have ... < r
                using m1 by auto
              finally show ?thesis .
            qed
          qed
        qed
      qed(use NiD sep in auto)
    qed
  qed
qed

lemma separable-LPm:
  assumes separable-space mtopology
  shows separable-space LPm.mtopology
proof(cases M = {})
  case True
  from M-empty-P[OF this] show ?thesis
    by(intro countable-space-separable-space) auto
next
  case M-ne:False
  then obtain ai :: nat ⇒ 'a where ai:mdense (range ai)
    using assms mdense-empty-iff uncountable-def unfolding separable-space-def2
by blast
  have countable (((λ(k, bi). sum-measure (borel-of mtopology) {..k}
                                 (λi. scale-measure (ennreal (bi i)) (return (borel-of
mtopology) (ai i))))
```

' (*SIGMA k*:*UNIV*. {..*k*} $\rightarrow_E$ $\mathbb{Q}$ ∩ {*0*..})))
using *countable-rat* **by**(*auto intro*!: *countable-PiE*)
thus *?thesis*
using *LPm-countable-base*[*OF ai*] **by**(*auto simp*: *separable-space-def2*)
qed

lemma *closedin-bounded-measures*:
closedin LPm.mtopology {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *N* (*space N*) ≤ *ennreal r*}
unfolding *LPm.metric-closedin-iff-sequentially-closed*
proof(*intro allI conjI uncurry impI*)
show *1*: {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *emeasure N* (*space N*) ≤ *ennreal r*} ⊆ $\mathcal{P}$
by(*auto intro*!: *inP-I finite-measureI simp*: *top.extremum-unique*)
fix *Ni N*
assume *h*:*range Ni* ⊆ {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *emeasure N* (*space N*) ≤ *ennreal r*}
limitin LPm.mtopology Ni N sequentially
then have *sets-Ni*: ⋀*i*. *sets* (*Ni i*) = *sets* (*borel-of mtopology*)
and *Nir*:⋀*i*. *Ni i* (*space* (*Ni i*)) ≤ *ennreal r*
by *auto*
interpret *N*: *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology* **by**(*simp add*: $\mathcal{P}$*-def*)
interpret *Ni*: *finite-measure Ni i* **for** *i*
using *1 h* **by**(*auto dest*: *inP-D*)
have *mweak-conv Ni N sequentially*
using *h 1 sets-Ni Nir* **by**(*auto intro*!: *converge-imp-mweak-conv*)
hence ⋀*f*. *continuous-map mtopology euclideanreal f*
⟹ (∃ *B*. ∀ *x*∈*M*. |*f x*| ≤ *B*) ⟹ (λ*n*. ∫ *x*. *f x* ∂*Ni n*) ⟶ (∫ *x*. *f x* ∂*N*)
by(*simp add*: *mweak-conv-def*)
from *this*[*of* λ*x*. *1*] **have** (λ*i*. *measure* (*Ni i*) (*space* (*Ni i*))) ⟶ *measure N* (*space N*)
by *auto*
hence (λ*i*. *Ni i* (*space* (*Ni i*))) ⟶ *N* (*space N*)
by (*simp add*: *N.emeasure-eq-measure Ni.emeasure-eq-measure*)
from *tendsto-upperbound*[*OF this*,*of ennreal r*]
show *N* ∈ {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *emeasure N* (*space N*) ≤ *ennreal r*}
using *limitin-topspace*[*OF h*(*2*)] *Nir* **unfolding** *LPm.topspace-mtopology*
by(*auto simp*: $\mathcal{P}$*-def*)
qed

lemma *closedin-subprobs*:
closedin LPm.mtopology {*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)}
unfolding *LPm.metric-closedin-iff-sequentially-closed*
proof(*intro allI conjI uncurry impI*)

show *1*:{*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)} ⊆ 𝒫
by(*auto intro*!: *inP-I simp*: *top.extremum-unique subprob-space-def*)
fix *Ni N*
assume *h*:*range Ni* ⊆ {*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)}
limitin LPm.mtopology Ni N sequentially
then have *sets-Ni*: ⋀*i*. *sets* (*Ni i*) = *sets* (*borel-of mtopology*) **and** *Ni*:⋀*i*. *subprob-space* (*Ni i*)
by *auto*
have *setsN*:*sets N* = *sets* (*borel-of mtopology*)
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology* **by**(*auto dest*: *inP-D*)
interpret *N*: *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology* **by**(*simp add*: 𝒫*-def*)
interpret *Ni*: *subprob-space Ni i* **for** *i*
by *fact*
have *mweak-conv Ni N sequentially*
using *h 1 sets-Ni Ni* **by**(*auto intro*!: *converge-imp-mweak-conv*)
hence ⋀*f*. *continuous-map mtopology euclideanreal f* ⟹ (∃ *B*. ∀ *x*∈*M*. |*f x*| ≤ *B*)
⟹ (λ*n*. ∫ *x*. *f x* ∂*Ni n*) ⟶ (∫ *x*. *f x* ∂*N*)
by(*simp add*: *mweak-conv-def*)
from *this*[*of* λ*x*. *1*] **have** (λ*i*. *measure* (*Ni i*) (*space* (*Ni i*))) ⟶ *measure N* (*space N*)
by *auto*
hence (λ*i*. *Ni i* (*space* (*Ni i*))) ⟶ *N* (*space N*)
by (*simp add*: *N.emeasure-eq-measure Ni.emeasure-eq-measure*)
from *tendsto-upperbound*[*OF this*,*of 1*]
have *emeasure N* (*space N*) ≤ *1*
using *Ni.subprob-emeasure-le-1* **by** *force*
moreover have *space N* ≠ {}
using *sets-eq-imp-space-eq*[*OF setsN*] *sets-eq-imp-space-eq*[*OF sets-Ni*[*of 0*]]
using *Ni.subprob-not-empty* **by** *fastforce*
ultimately show *N* ∈ {*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)}
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology*
by(*auto intro*!: *subprob-spaceI setsN*)
qed

lemma *closedin-probs*: *closedin LPm.mtopology* {*N*. *prob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)}
unfolding *LPm.metric-closedin-iff-sequentially-closed*
proof(*intro allI conjI uncurry impI*)
show *1*:{*N*. *prob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)} ⊆ 𝒫
by(*auto intro*!: *inP-I simp*: *top.extremum-unique prob-space-def*)
fix *Ni N*
assume *h*:*range Ni* ⊆ {*N*. *prob-space N* ∧ *sets N* = *sets* (*borel-of mtopology*)}
limitin LPm.mtopology Ni N sequentially
then have *sets-Ni*: ⋀*i*. *sets* (*Ni i*) = *sets* (*borel-of mtopology*) **and** *Ni*:⋀*i*. *prob-space* (*Ni i*)

by *auto*
have *setsN*:*sets N* = *sets* (*borel-of mtopology*)
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology* **by**(*auto dest*: *inP-D*)
interpret *N*: *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology* **by**(*simp add*: $\mathcal{P}$-*def*)
interpret *Ni*: *prob-space Ni i* **for** *i*
by *fact*
have *mweak-conv Ni N sequentially*
using *h 1 sets-Ni Ni* **by**(*auto intro*!: *converge-imp-mweak-conv*)
hence $\bigwedge f.$ *continuous-map mtopology euclideanreal f* $\Longrightarrow$ ($\exists B.\ \forall x{\in}M.\ |f\ x| \le B$)
$\Longrightarrow (\lambda n.\ \int x.\ f\ x\ \partial Ni\ n) \longrightarrow (\int x.\ f\ x\ \partial N)$
by(*simp add*: *mweak-conv-def*)
from *this*[*of* $\lambda x.$ *1*] **have** ($\lambda i.$ *measure* (*Ni i*) (*space* (*Ni i*))) $\longrightarrow$ *measure N* (*space N*)
by *auto*
hence *prob-space N*
by(*simp add*: *Ni.prob-space LIMSEQ-const-iff N.emeasure-eq-measure prob-spaceI*)
thus $N \in \{N.$ *prob-space N* $\wedge$ *sets N* = *sets* (*borel-of mtopology*)$\}$
using *limitin-topspace*[*OF h*(*2*)] **unfolding** *LPm.topspace-mtopology*
by(*auto intro*!: *setsN*)
qed

4.4 The Lévy-Prokhorov Metric and Topology of Weak Convegence

lemma *weak-conv-topology-le-LPm-topology*:
assumes*openin* (*weak-conv-topology mtopology*) *S*
shows *openin LPm.mtopology S*
proof(*rule weak-conv-topology-minimal*[*OF - - assms*])
fix *f B*
assume *f*: *continuous-map mtopology euclideanreal f* **and** *B*:$\bigwedge x.\ x \in$ *topspace mtopology* $\Longrightarrow |f\ x| \le B$
show *continuous-map LPm.mtopology euclideanreal* ($\lambda N.\ \int x.\ f\ x\ \partial N$)
unfolding *continuous-map-iff-limit-seq*[*OF LPm.first-countable-mtopology*]
proof *safe*
fix *Ni N*
assume *limitin LPm.mtopology Ni N sequentially*
then have *h'*:*weak-conv-on Ni N sequentially mtopology*
by(*simp add*: *mtopology-of-def converge-imp-mweak-conv*)
thus *limitin euclideanreal* ($\lambda n.\ \int x.\ f\ x\ \partial Ni\ n$) ($\int x.\ f\ x\ \partial N$) *sequentially*
using *f B* **by**(*fastforce simp*: *mweak-conv-seq-def*)
qed
qed(*unfold LPm.topspace-mtopology*, *simp add*: $\mathcal{P}$-*def*)

lemma *LPmtopology-eq-weak-conv-topology*:
assumes *separable-space mtopology*

shows *LPm.mtopology* = *weak-conv-topology mtopology*
by(*auto intro*!: *topology-eq-filter inP-I simp*: *conv-iff-mweak-conv*[*OF assms*] *inP-D*)

end

corollary
assumes *metrizable-space X separable-space X*
shows *metrizable-weak-conv-topology*:*metrizable-space* (*weak-conv-topology X*)
and *separable-weak-conv-topology*:*separable-space* (*weak-conv-topology X*)
proof −
obtain *d* **where** *d*:*Metric-space* (*topspace X*) *d Metric-space.mtopology* (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms*(*1*) *metrizable-space-def*)
then interpret *Levy-Prokhorov topspace X d*
by(*auto simp*: *Levy-Prokhorov-def*)
show *g1*:*metrizable-space* (*weak-conv-topology X*)
using *assms*(*2*) *d*(*2*) *LPm.metrizable-space-mtopology LPmtopology-eq-weak-conv-topology* **by** *simp*
show *g2*:*separable-space* (*weak-conv-topology X*)
using *assms*(*2*) *d*(*2*) *LPmtopology-eq-weak-conv-topology separable-LPm* **by** *simp*
qed

end

5 Prokhorov's Theorem

theory *Prokhorov-Theorem*
imports *Levy-Prokhorov-Distance*
Alaoglu-Theorem
begin

5.1 Prokhorov's Theorem

context *Levy-Prokhorov*
begin

lemma *relatively-compact-imp-tight-LP*:
assumes $\Gamma \subseteq \mathcal{P}$ *separable-space mtopology mcomplete*
and *compactin LPm.mtopology* (*LPm.mtopology closure-of* Γ)
shows *tight-on-set mtopology* Γ
proof(*cases M* = {})
case *True*
then have $\Gamma = \{\} \vee \Gamma = \{$*null-measure* (*borel-of mtopology*)$\}$
using *assms*(*1*) *M-empty-P'* **by** *auto*
thus *?thesis*
by(*auto simp*: *tight-on-set-def intro*!: *finite-measureI*)
next
case *M-ne*:*False*

have *1*: ∃*k*. ∀*N*∈Γ. *measure N* (⋃*m*≤*k*. *Ui m*) > *measure N M* − *e*
if *Ui*: ⋀*i*::*nat*. *openin mtopology* (*Ui i*) (⋃*i*. *Ui i*) = *M* **and** *e*:*e* > *0* **for** *Ui e*
proof(*rule ccontr*)
assume ∄*k*. ∀*N*∈Γ. *measure N* (⋃*m*≤*k*. *Ui m*) > *measure N M* − *e*
then have *h*: ∀*k*. ∃*N*∈Γ. *measure N* (⋃*m*≤*k*. *Ui m*) ≤ *measure N M* − *e*
by(*auto simp*: *linorder-class.not-less*)
then obtain *Nk* **where** *Nk*: ⋀*k*. *Nk k* ∈ Γ ⋀*k*. *measure* (*Nk k*) (⋃*m*≤*k*. *Ui m*) ≤ *measure* (*Nk k*) *M* − *e*
by *metis*
obtain *N r* **where** *Nr*: *N* ∈ *LPm.mtopology closure-of* Γ *strict-mono r*
limitin LPm.mtopology (*Nk* ∘ *r*) *N sequentially*
using *assms*(*1*,*4*) *Nk*(*1*) *closure-of-subset*[*of* Γ *LPm.mtopology*]
by(*simp add*: *LPm.compactin-sequentially*) (*metis image-subset-iff subsetD*)
then interpret *mweak-conv-fin M d* λ*i*. *Nk* (*r i*) *N sequentially*
using *assms*(*1*) *Nk*(*1*) *closure-of-subset-topspace*[*of LPm.mtopology*]
by(*auto intro*!: *inP-mweak-conv-fin-all*)
have *sets-Nk*[*measurable-cong*,*simp*]:⋀*i*. *sets* (*Nk* (*r i*)) = *sets* (*borel-of mtopology*)
using *Nk*(*1*) *assms*(*1*) *inP-D*(*2*) **by** *blast*
have *wc*: *mweak-conv-seq* (λ*i*. *Nk* (*r i*)) *N*
using *converge-imp-mweak-conv*[*OF Nr*(*3*)] *Nk*(*1*) *assms*(*1*) **by**(*auto simp*: *comp-def*)
interpret *Nk*: *finite-measure Nk k* **for** *k*
using *Nk*(*1*) *assms*(*1*) *inP-D* **by** *blast*
interpret *N*: *finite-measure N*
using *finite-measure-N* **by** *blast*
have *1*:*measure N* (⋃*i*≤*n*. *Ui i*) ≤ *measure N M* − *e* **for** *n*
proof −
have *measure N* (⋃*i*≤*n*. *Ui i*) ≤ *liminf* (λ*j*. *measure* (*Nk* (*r j*)) (⋃*i*≤*n*. *Ui i*))
using *Ui* **by**(*auto intro*!: *conjunct2*[*OF mweak-conv-eq3*[*THEN iffD1*,*OF wc*],*rule-format*])
also have ... ≤ *liminf* (λ*j*. *measure* (*Nk* (*r j*)) (⋃*i*≤*r j*. *Ui i*))
by(*rule Liminf-mono*)
(*auto intro*!: *Ui*(*1*) *exI*[**where** *x*=*n*] *Nk.finite-measure-mono*[*OF UN-mono*]
le-trans[*OF - strict-mono-imp-increasing*[*OF Nr*(*2*)]] *borel-of-open*
simp: *eventually-sequentially sets-N*)
also have ... ≤ *liminf* (λ*j*. *measure* (*Nk* (*r j*)) *M* + *ereal* (− *e*))
using *Nk* **by**(*auto intro*!: *Liminf-mono eventuallyI*)
also have ... ≤ *liminf* (λ*j*. *measure* (*Nk* (*r j*)) *M*) + *limsup* (λ*i*. − *e*)
by(*rule ereal-liminf-limsup-add*)
also have ... = *liminf* (λ*j*. *measure* (*Nk* (*r j*)) *M*) + *ereal* (− *e*)
using *Limsup-const*[*of sequentially* − *e*] **by** *simp*
also have ... = *measure N M* + *ereal* (− *e*)
proof −
have (λ*k*. *measure* (*Nk* (*r k*)) *M*) ⟶ *measure N M*
using *wc mweak-conv-eq2* **by** *fastforce*
from *limI*[*OF tendsto-ereal*[*OF this*]] *convergent-liminf-cl*[*OF convergentI*[*OF tendsto-ereal*[*OF this*]]]

show *?thesis* **by** *simp*
qed
finally show *?thesis*
by *simp*
qed
have *2*:(λn. *measure N* ($\bigcup i\leq n$. *Ui i*)) $\longrightarrow$ *measure N M*
proof −
have (λn. *measure N* ($\bigcup i\leq n$. *Ui i*)) $\longrightarrow$ *measure N* ($\bigcup$ (*range* (λn. $\bigcup i\leq n$. *Ui i*)))
by(*fastforce intro*!: *Ui*(*1*) *N.finite-Lim-measure-incseq borel-of-open incseq-SucI simp*: *sets-N*)
moreover have $\bigcup$ (*range* (λn. $\bigcup i\leq n$. *Ui i*)) = *M*
using *Ui*(*2*) **by** *blast*
ultimately show *?thesis*
by *simp*
qed
show *False*
using *e Lim-bounded*[*OF 2*,*of 0 measure N M* − *e*] *1* **by** *auto*
qed
show *?thesis*
unfolding *tight-on-set-def*
proof *safe*
fix *e* :: *real*
assume *e*: *0* < *e*
obtain *U* **where** *U*: *countable U mdense U*
using *assms*(*2*) *separable-space-def2* **by** *blast*
let *?an* = *from-nat-into U*
have *an*: $\bigwedge n$. *?an n* ∈ *M mdense* (*range ?an*)
by (*metis M-ne U*(*2*) *from-nat-into mdense-def2 mdense-empty-iff subsetD*)
(*metis M-ne U*(*1*) *U*(*2*) *mdense-empty-iff range-from-nat-into*)
have ∃ *k*. ∀ *N*∈Γ. *measure N* ($\bigcup n\leq k$. *mball* (*?an n*) (*1* / *Suc m*)) > *measure N M* − (*e* / *2*) ∗ (*1* / *2*) ^ *Suc m* **for** *m*
by(*rule 1*) (*use mdense-balls-cover*[*OF an*(*2*)] *e* **in** *auto*)
then obtain *k* **where** *k*:
$\bigwedge m$ *N*. *N* ∈ Γ $\Longrightarrow$ *measure N* ($\bigcup n\leq k$ *m*. *mball* (*?an n*) (*1* / *Suc m*)) > *measure N M* − (*e* / *2*) ∗ (*1* / *2*) ^ *Suc m*
by *metis*
let *?K* = $\bigcap m$. ($\bigcup i\leq k$ *m*. *mcball* (*?an i*) (*1* / *Suc m*))
show ∃ *K*. *compactin mtopology K* ∧ (∀ *M*∈Γ. *measure M* (*space M* − *K*) < *e*)
proof(*safe intro*!: *exI*[**where** *x*=*?K*])
have *closedin mtopology ?K*
by(*auto intro*!: *closedin-Union*)
moreover have *?K* ⊆ *M*
by *auto*
moreover have *mtotally-bounded ?K*
unfolding *mtotally-bounded-def2*
proof *safe*
fix *e* :: *real*
assume *e*: *0* < *e*

then obtain *m* **where** *m*: *e > 1 / Suc m*
using *nat-approx-posE* **by** *blast*
have *?K* ⊆ (⋃ *i≤k m. mcball (?an i) (1 / real (Suc m))*)
by *auto*
also have ... ⊆ (⋃ *x∈?an ‘ {..k m}. mball x e*)
using *mcball-subset-mball-concentric*[*OF m*] **by** *blast*
finally show ∃ *K. finite K* ∧ *K* ⊆ *M* ∧ *?K* ⊆ (⋃ *x∈K. mball x e*)
using *an*(*1*) **by**(*fastforce intro*!: *exI*[**where** *x=?an ‘ {..k m}*])
qed
ultimately show *compactin mtopology ?K*
using *mtotally-bounded-eq-compact-closedin*[*OF assms*(*3*)] **by** *auto*
next
fix *N*
assume *N*: *N* ∈ Γ
then interpret *N*: *finite-measure N*
using *assms*(*1*) *inP-D* **by** *blast*
have *sets-N*: *sets N = sets* (*borel-of mtopology*)
using *N assms*(*1*) **by**(*auto simp*: 𝒫*-def*)
hence *space-N*: *space N = M*
by(*auto cong*: *sets-eq-imp-space-eq simp*: *space-borel-of*)
have [*measurable*]:⋀*a b. mcball a b* ∈ *sets N M* ∈ *sets N*
by(*auto simp*: *sets-N intro*!: *borel-of-closed*)
have *Ne*:*measure N* (*M* − (⋃ *i≤k m. mcball (?an i) (1 / real (Suc m)*))) < (*e / 2*) ∗ (*1 / 2*) ^ *Suc m* **for** *m*
proof −
have *measure N* (*M* − (⋃ *i≤k m. mcball (?an i) (1 / real (Suc m)*)))
= *measure N M* − *measure N* (⋃ *i≤k m. mcball (?an i) (1 / real (Suc m)*))
by(*auto simp*: *N.finite-measure-compl*[*simplified space-N*])
also have ... ≤ *measure N M* − *measure N* (⋃ *i≤k m. mball (?an i) (1 / real (Suc m)*))
by(*fastforce intro*!: *N.finite-measure-mono*)
also have ... < (*e / 2*) ∗ (*1 / 2*) ^ *Suc m*
using *k*[*OF N*,*of m*] **by** *simp*
finally show *?thesis* **.**
qed
have *Ne-sum*: *summable* (λ*m.* (*e / 2*) ∗ (*1 / 2*) ^ *Suc m*)
by *auto*
have *sum2*: *summable* (λ*m. measure N* (*M* − (⋃ *i≤k m. mcball (from-nat-into U i) (1 / real (Suc m)*)))))
using *Ne* **by**(*auto intro*!: *summable-comparison-test-ev*[*OF - Ne-sum*] *eventuallyI*) (*use less-eq-real-def* **in** *blast*)
show *measure N* (*space N* − *?K*) < *e*
proof −
have *measure N* (*space N* − *?K*) = *measure N* (⋃ *m.* (*M* − (⋃ *i≤k m. mcball (?an i) (1 / Suc m)*))))
by(*auto simp*: *space-N*)
also have ... ≤ (∑ *m. measure N* (*M* − (⋃ *i≤k m. mcball (?an i) (1 / Suc m)*))))

```
        by(rule N.finite-measure-subadditive-countably) (use sum2 in auto)
      also have ... ≤ (∑ m. (e / 2) * (1 / 2) ^ Suc m)
        by(rule suminf-le) (use Ne less-eq-real-def sum2 in auto)
      also have ... = (e / 2) * (∑ m. (1 / 2) ^ Suc m)
        by(rule suminf-mult) auto
      also have ... = e / 2
        using power-half-series sums-unique by fastforce
      also have ... < e
        using e by simp
      finally show ?thesis .
    qed
  qed
 qed(use assms inP-D in auto)
qed

lemma mcompact-imp-LPmcompact:
 assumes compact-space mtopology
  shows compactin LPm.mtopology {N. sets N = sets (borel-of mtopology) ∧ N
(space N) ≤ ennreal r}
   (is compactin - ?N)
proof −
 consider M = {} | r < 0 | r ≥ 0 M ≠ {}
   by linarith
 then show ?thesis
 proof cases
   assume M = {}
   then have finite (topspace LPm.mtopology)
     unfolding LPm.topspace-mtopology using M-empty-P by fastforce
   thus ?thesis
      using closedin-bounded-measures closedin-compact-space compact-space-def
finite-imp-compactin-eq by blast
 next
   assume r < 0
   then have ?N = {null-measure (borel-of mtopology)}
     using emeasure-eq-0[OF - - sets.sets-into-space]
     by(safe,intro measure-eqI) (auto simp: ennreal-lt-0)
   thus ?thesis
     by(auto intro!: inP-I finite-measureI)
 next
   assume M-ne:M ≠ {} and r:r ≥ 0
   hence [simp]: mtopology ≠ trivial-topology
     using topspace-mtopology by force
   define Cb where Cb ≡ cfunspace mtopology (euclidean-metric :: real metric)
  define Cb' where Cb' ≡ powertop-real (mspace (cfunspace mtopology (euclidean-metric
:: real metric)))
   define B where
   B ≡ {φ∈topspace Cb'. φ (λx∈topspace mtopology. 1) ≤ r ∧ positive-linear-functional-on-CX
mtopology φ}
   define T :: 'a measure ⇒ ('a ⇒ real) ⇒ real
```

where *T* ≡ *λN*. *λf*∈*mspace* (*cfunspace mtopology euclidean-metric*). ∫ *x*. *f x* ∂*N*

have *compact*: *compactin Cb′ B*

unfolding *B-def Cb′-def* **by**(*rule Alaoglu-theorem-real-functional*[*OF assms*(*1*)]) (*use M-ne* **in** *simp*)

have *metrizable*: *metrizable-space* (*subtopology Cb′ B*)

unfolding *B-def Cb′-def* **by**(*rule metrizable-functional*[*OF assms metrizable-space-mtopology*])

have *homeo*: *homeomorphic-map* (*subtopology LPm.mtopology ?N*) (*subtopology Cb′ B*) *T*

proof −

have *T-cont′*: *continuous-map* (*subtopology LPm.mtopology ?N*) *Cb′ T*

unfolding *continuous-map-atin*

proof *safe*

fix *N*

assume *N*:*N* ∈ *topspace* (*subtopology LPm.mtopology ?N*)

show *limitin Cb′ T* (*T N*) (*atin* (*subtopology LPm.mtopology ?N*) *N*)

unfolding *Cb′-def limitin-componentwise*

proof *safe*

fix *g* :: ′*a* ⇒ *real*

assume *g*:*g* ∈ *mspace* (*cfunspace mtopology euclidean-metric*)

then have *g-bounded*:∃ *B*. ∀ *x*∈*M*. |*g x*| ≤ *B*

by(*auto simp*: *bounded-pos-less order-less-le*)

show *limitin euclideanreal* (*λc*. *T c g*) (*T N g*) (*atin* (*subtopology LPm.mtopology ?N*) *N*)

unfolding *limitin-canonical-iff*

proof

fix *e* :: *real*

assume *e*:*0* < *e*

have *N-in*: *N* ∈ *?N*

using *N* **by** *simp*

show ∀ *F* *c in atin* (*subtopology LPm.mtopology ?N*) *N*. *dist* (*T c g*) (*T N g*) < *e*

unfolding *atin-subtopology-within*[*OF N-in*]

proof (*safe intro*!: *eventually-within-imp*[*THEN iffD2*,*OF LPm.eventually-atin-sequentially*[*THEN iffD2*]])

fix *Ni*

assume *Ni*:*range Ni* ⊆ 𝒫 − {*N*} *limitin LPm.mtopology Ni N sequentially*

with *N* **interpret** *mweak-conv-fin M d Ni N sequentially*

by(*auto intro*!: *inP-mweak-conv-fin-all*)

have *wc*:*mweak-conv-seq Ni N*

using *Ni* **by**(*auto intro*!: *converge-imp-mweak-conv*)

hence *1*:(*λn*. *T* (*Ni n*) *g*) ⟶ *T N g*

unfolding *T-def* **by**(*auto simp*: *g mweak-conv-def g-bounded*)

show ∀ *F* *n in sequentially*. *Ni n* ∈ *?N* ⟶ *dist* (*T* (*Ni n*) *g*) (*T N g*) < *e*

by(*rule eventually-mp*[*OF - 1*[*simplified tendsto-iff*,*rule-format*,*OF e*]]) *simp*

qed

qed
qed(*auto simp*: *T-def*)
qed
have *T-cont*: *continuous-map* (*subtopology LPm.mtopology ?N*) (*subtopology Cb′ B*) *T*
unfolding *continuous-map-in-subtopology*
proof
show *T ‘ topspace* (*subtopology LPm.mtopology ?N*) ⊆ *B*
unfolding *B-def Cb′-def*
proof *safe*
fix *N*
assume *N*:*N* ∈ *topspace* (*subtopology LPm.mtopology ?N*)
then have *finite-measure N* **and** *sets-N*:*sets N* = *sets* (*borel-of mtopology*)
and *space-N*:*space N* = *M* **and** *N-r*:*emeasure N* (*space N*) ≤ *ennreal r*
by(*auto intro*!: *inP-D*)
hence *N-r′*:*measure N* (*space N*) ≤ *r*
by (*simp add*: *finite-measure.emeasure-eq-measure r*)
interpret *N*: *finite-measure N*
by *fact*
have *TN-def*: *T N* (λ*x*∈*topspace mtopology*. *f x*) = (∫ *x*. *f x* ∂*N*) *T N* (λ*x*∈*M*. *f x*) = (∫ *x*. *f x* ∂*N*)
if *f*:*continuous-map mtopology euclideanreal f* **for** *f*
using *f Bochner-Integration.integral-cong*[*OF refl*,*of N* λ*x*∈*M*. *f x f*,*simplified space-N*]
compact-imp-bounded[*OF compactin-euclidean-iff* [*THEN iffD1*,
OF image-compactin[*OF assms*[*simplified compact-space-def*] *f*]]]
by(*auto simp*: *T-def*)
have *N-integrable*[*simp*]: *integrable N f* **if** *f*:*continuous-map mtopology euclideanreal f* **for** *f*
using *compact-imp-bounded*[*OF compactin-euclidean-iff* [*THEN iffD1*,*OF image-compactin*[*OF*
assms[*simplified compact-space-def*] *f*]]] *continuous-map-measurable*[*OF f*]
by(*auto intro*!: *N.integrable-const-bound AE-I2*[*of N*]
simp: *bounded-iff measurable-cong-sets*[*OF sets-N*] *borel-of-euclidean space-N*)

show *T N* (λ*x*∈*topspace mtopology*. *1*) ≤ *r*
unfolding *TN-def*[*OF continuous-map-canonical-const*]
using *N-r′* **by** *simp*
show *positive-linear-functional-on-CX mtopology* (*T N*)
unfolding *positive-linear-functional-on-CX-compact*[*OF assms*]
proof *safe*
fix *f c*
assume *f*: *continuous-map mtopology euclideanreal f*
show *T N* (λ*x*∈*topspace mtopology*. *c* ∗ *f x*) = *c* ∗ *T N* (λ*x*∈*topspace mtopology*. *f x*)
using *f continuous-map-real-mult-left*[*OF f*,*of c*] **by**(*auto simp*: *TN-def*)
next

```
fix f g
assume fg: continuous-map mtopology euclideanreal f
  continuous-map mtopology euclideanreal g
show T N (λx∈topspace mtopology. f x + g x)
    = T N (λx∈topspace mtopology. f x) + T N (λx∈topspace mtopology.
g x)
  using fg continuous-map-add[OF fg]
  by(auto simp: TN-def intro!: Bochner-Integration.integral-add)
next
  fix f
 assume continuous-map mtopology euclideanreal f ∀ x∈topspace mtopology.
0 ≤ f x
  then show 0 ≤ T N (λx∈topspace mtopology. f x)
 by(auto simp: TN-def space-N intro!: Bochner-Integration.integral-nonneg)
qed
  show T N ∈ topspace (powertop-real (mspace (cfunspace mtopology
euclidean-metric)))
  by(auto simp: T-def)
qed
qed fact
define T-inv :: (('a ⇒ real) ⇒ real) ⇒ 'a measure where
  T-inv ≡ (λφ. THE N. sets N = sets (borel-of mtopology) ∧ finite-measure
N ∧
                   (∀ f. continuous-map mtopology euclideanreal f
                      ⟶ φ (restrict f (topspace mtopology)) = integral^L N f))
have T-T-inv: ∀ N∈topspace (subtopology LPm.mtopology ?N). T-inv (T N)
= N
proof safe
  fix N
  assume N:N ∈ topspace (subtopology LPm.mtopology ?N)
  from Pi-mem[OF continuous-map-funspace[OF T-cont] this]
  have TN:T N ∈ topspace (subtopology Cb' B)
    by blast
  hence ∃!N'. sets N' = sets (borel-of mtopology) ∧ finite-measure N' ∧
          (∀ f. continuous-map mtopology euclideanreal f
              ⟶ T N (restrict f (topspace mtopology)) = integral^L N' f)
    by(intro Riesz-representation-real-compact-metrizable[OF assms metriz-
able-space-mtopology])
      (auto simp del: topspace-mtopology restrict-apply simp: B-def)
  moreover have sets N = sets (borel-of mtopology) ∧ finite-measure N ∧
          (∀ f. continuous-map mtopology euclideanreal f
              ⟶ T N (restrict f (topspace mtopology)) = integral^L N f)
    using compact-imp-bounded[OF compactin-euclidean-iff[THEN iffD1,OF
image-compactin[OF
          assms[simplified compact-space-def] -]]] N
    by(auto simp: T-def dest:inP-D cong: Bochner-Integration.integral-cong)
  ultimately show T-inv (T N) = N
    unfolding T-inv-def by(rule the1-equality)
qed
```

have *T-inv-T*: ∀ φ∈*topspace* (*subtopology Cb′ B*). *T* (*T-inv* φ) = φ
proof *safe*
fix φ
assume φ:φ∈*topspace* (*subtopology Cb′ B*)
hence *1*:∃!*N′*. *sets N′* = *sets* (*borel-of mtopology*) ∧ *finite-measure N′* ∧
(∀ *f*. *continuous-map mtopology euclideanreal f*
⟶ φ (*restrict f* (*topspace mtopology*)) = *integral*^{L} *N′ f*)
by(*intro Riesz-representation-real-compact-metrizable*[*OF assms metrizable-space-mtopology*])
(*auto simp del*: *topspace-mtopology restrict-apply simp add*: *B-def*)
have *T-inv-*φ:*sets* (*T-inv* φ) = *sets* (*borel-of mtopology*) *finite-measure* (*T-inv* φ)
⋀*f*. *continuous-map mtopology euclideanreal f*
⟹ φ (λ*x*∈*topspace mtopology*. *f x*) = *integral*^{L} (*T-inv* φ) *f*
unfolding *T-inv-def* **by**(*use theI′*[*OF 1*] **in** *blast*)+
show *T* (*T-inv* φ) = φ
proof
fix *f*
consider *f* ∈ *mspace Cb* | *f* ∉ *mspace Cb*
by *fastforce*
then show *T* (*T-inv* φ) *f* = φ *f*
proof *cases*
case *1*
then have *T* (*T-inv* φ) *f* = *integral*^{L} (*T-inv* φ) *f*
by(*auto simp*: *T-def Cb-def*)
also have ... = φ (λ*x*∈*topspace mtopology*. *f x*)
by(*rule T-inv-*φ(*3*)[*symmetric*]) (*use 1 Cb-def* **in** *auto*)
also have ... = φ *f*
proof −
have *2*: (λ*x*∈*topspace mtopology*. *f x*) = *f*
using *1* **by**(*auto simp*: *extensional-def Cb-def*)
show *?thesis*
unfolding *2* **by** *blast*
qed
finally show *?thesis* **.**
next
case *2*
then have *T* (*T-inv* φ) *f* = *undefined*
by (*auto simp*: *Cb-def T-def*)
also have ... = φ *f*
using *2* φ *Cb′-def Cb-def PiE-arb* **by** *auto*
finally show *?thesis* **.**
qed
qed
qed
have *T-inv-cont*: *continuous-map* (*subtopology Cb′ B*) (*subtopology LPm.mtopology ?N*) *T-inv*
unfolding *seq-continuous-iff-continuous-first-countable*[*OF metrizable-imp-first-countable*[*OF metrizable*],*symmetric*] *seq-continuous-map*

proof *safe*
fix φn φ
assume *limitin* (*subtopology Cb′ B*) φn φ *sequentially*
then have φB: $\varphi \in B$ **and** *h*:*limitin Cb′* φn φ *sequentially* $\forall_F$ *n in sequentially*. φn *n* $\in B$
by(*auto simp*: *limitin-subtopology*)
then obtain *n0* **where** *n0*: $\bigwedge n.\ n \geq n0 \Longrightarrow \varphi n\ n \in B$
by(*auto simp*: *eventually-sequentially*)
have *limit*: $\bigwedge f.\ f \in$ *mspace* (*cfunspace mtopology euclidean-metric*) $\Longrightarrow$ (λn. φn *n f*) $\longrightarrow$ φ *f*
using *h*(*1*) **by**(*auto simp*: *limitin-componentwise Cb′-def*)
show *limitin* (*subtopology LPm.mtopology ?N*) (λn. *T-inv* (φn *n*)) (*T-inv* φ) *sequentially*
proof(*rule limitin-sequentially-offset-rev*[**where** *k*=*n0*])
from φB **have** $\exists!N$. *sets N* = *sets* (*borel-of mtopology*) $\wedge$ *finite-measure N* $\wedge$
($\forall f$. *continuous-map mtopology euclideanreal f*
$\longrightarrow$ φ (*restrict f* (*topspace mtopology*)) = *integral*^{L} *N f*)
by(*intro Riesz-representation-real-compact-metrizable*[*OF assms metrizable-space-mtopology*])
(*auto simp del*: *topspace-mtopology restrict-apply simp*: *B-def*)
hence *sets* (*T-inv* φ) = *sets* (*borel-of mtopology*) $\wedge$ *finite-measure* (*T-inv* φ) $\wedge$
($\forall f$. *continuous-map mtopology euclideanreal f*
$\longrightarrow$ φ (*restrict f* (*topspace mtopology*)) = *integral*^{L} (*T-inv* φ) *f*)
unfolding *T-inv-def* **by**(*rule theI′*)
hence *T-inv-*φ: *sets* (*T-inv* φ) = *sets* (*borel-of mtopology*) *finite-measure* (*T-inv* φ)
$\bigwedge f$. *continuous-map mtopology euclideanreal f*
$\Longrightarrow$ φ (*restrict f* (*topspace mtopology*)) = *integral*^{L} (*T-inv* φ) *f*
by *auto*
from *this*(*2*) *this*(*3*)[*of* λx. *1*] φB **have** *T-inv-*φ*-r*: *T-inv* φ (*space* (*T-inv* φ)) $\leq$ *ennreal r*
unfolding *B-def* **by** *simp* (*metis ennreal-le-iff finite-measure.emeasure-eq-measure r*)
{
fix *n*
from *n0*[*of n* + *n0*,*simplified*] **have** $\exists!N$. *sets N* = *sets* (*borel-of mtopology*) $\wedge$
finite-measure N $\wedge$ ($\forall f$. *continuous-map mtopology euclideanreal f*
$\longrightarrow$ φn (*n* + *n0*) (*restrict f* (*topspace mtopology*)) = *integral*^{L} *N f*)
by(*intro Riesz-representation-real-compact-metrizable*[*OF assms metrizable-space-mtopology*])
(*auto simp del*: *topspace-mtopology restrict-apply simp*: *B-def*)
hence *sets* (*T-inv* (φn (*n* + *n0*))) = *sets* (*borel-of mtopology*) $\wedge$ *finite-measure* (*T-inv* (φn (*n* + *n0*))) $\wedge$
($\forall f$. *continuous-map mtopology euclideanreal f*
$\longrightarrow$ φn (*n* + *n0*) (*restrict f* (*topspace mtopology*)) = *integral*^{L} (*T-inv*

```
(φn (n + n0))) f)
            unfolding T-inv-def by(rule theI')
          hence sets (T-inv (φn (n + n0))) = sets (borel-of mtopology)
            finite-measure (T-inv (φn (n + n0)))
            ⋀f. continuous-map mtopology euclideanreal f
            ⟹ φn (n + n0) (restrict f (topspace mtopology)) = integral^L (T-inv
(φn (n + n0))) f
            by auto
        }
        note T-inv-φn = this
        have T-inv-φn-r: T-inv (φn (n + n0)) (space (T-inv (φn (n + n0)))) ≤
ennreal r for n
         using T-inv-φn(2)[of n] T-inv-φn(3)[of λx. 1 n] n0[of n + n0,simplified]
       unfolding B-def by simp (metis ennreal-le-iff finite-measure.emeasure-eq-measure
r)
       show limitin (subtopology LPm.mtopology ?N) (λn. T-inv (φn (n + n0)))
(T-inv φ) sequentially
          proof(intro limitin-subtopology[THEN iffD2] mweak-conv-imp-converge
conjI)
          show mweak-conv-seq (λn. T-inv (φn (n + n0))) (T-inv φ)
            unfolding mweak-conv-seq-def
          proof safe
            fix f :: 'a ⇒ real and B
            assume f:continuous-map mtopology euclideanreal f and B:∀x∈M. |f
x| ≤ B
           hence f': restrict f (topspace mtopology) ∈ mspace (cfunspace mtopology
euclidean-metric)
               by (auto simp: bounded-pos-less intro!: exI[where x=|B| + 1])
               have 1:(λn. ∫x. f x ∂ T-inv (φn (n + n0))) = (λn. φn (n + n0)
(restrict f (topspace mtopology)))
               by(subst T-inv-φn(3)) (use f in auto)
             have 2:(∫x. f x ∂ T-inv φ) = φ (restrict f (topspace mtopology))
               by(subst T-inv-φ(3)) (use f in auto)
            show (λn. ∫x. f x ∂ T-inv (φn (n + n0))) ⟶ (∫x. f x ∂T-inv φ)
              unfolding 1 2 using limit[OF f'] LIMSEQ-ignore-initial-segment by
blast
          qed(use T-inv-φ(1,2) T-inv-φn(1,2) eventuallyI in auto)
        next
          show ∀F a in sequentially. T-inv (φn (a + n0)) ∈ ?N
            by (simp add: T-inv-φn(1) T-inv-φn-r)
        next
           show T-inv φ ∈ {N. sets N = sets (borel-of mtopology) ∧ emeasure N
(space N) ≤ ennreal r}
            using T-inv-φ(1) T-inv-φ-r by auto
      qed(use T-inv-φn(1) T-inv-φn-r T-inv-φ(1) T-inv-φ-r compact-space-imp-separable[OF
assms] in auto)
      qed
    qed
    show ?thesis
```

using *T-inv-cont T-cont T-T-inv T-inv-T*
by(*auto intro*!: *homeomorphic-maps-imp-map*[**where** *g*=*T-inv*] *simp*: *homeomorphic-maps-def*)
qed
show *?thesis*
using *homeomorphic-compact-space*[*OF homeomorphic-map-imp-homeomorphic-space*[*OF homeo*]]
compact-space-subtopology[*OF compact*] *LPm.closedin-metric closedin-bounded-measures compactin-subspace*
by *fastforce*
qed
qed

lemma *tight-imp-relatively-compact-LP*:
assumes $\Gamma \subseteq \{N.$ *sets* $N =$ *sets* (*borel-of mtopology*) $\wedge$ N (*space* N) $\leq$ *ennreal* $r\}$ *separable-space mtopology*
and *tight-on-set mtopology* Γ
shows *compactin LPm.mtopology* (*LPm.mtopology closure-of* Γ)
proof(*cases* $r < 0$)
assume $r < 0$
then have $*$:$\{N.$ *sets* $N =$ *sets* (*borel-of mtopology*) $\wedge$ N (*space* N) $\leq$ *ennreal* $r\} = \{$*null-measure* (*borel-of mtopology*)$\}$
using *emeasure-eq-0*[*OF - - sets.sets-into-space*]
by(*safe*,*intro measure-eqI*) (*auto simp*: *ennreal-lt-0*)
with *assms*(*1*) **have** $\Gamma = \{\} \vee \Gamma = \{$*null-measure* (*borel-of mtopology*)$\}$
by *auto*
hence *LPm.mtopology closure-of* $\Gamma = \{\} \vee$ *LPm.mtopology closure-of* $\Gamma = \{$*null-measure* (*borel-of mtopology*)$\}$
by (*metis* (*no-types*) $*$ *closedin-bounded-measures closure-of-empty closure-of-eq*)
thus *?thesis*
by(*auto intro*!: *inP-I finite-measureI*)
next
assume $\neg\ r < 0$
then have *r-nonneg*:$r \geq 0$
by *simp*
have *subst1*: $\Gamma \subseteq \mathcal{P}$
using *assms*(*1*) *linorder-not-le* **by**(*force intro*!: *finite-measureI inP-I*)
have *subst2*: *LPm.mtopology closure-of* $\Gamma \subseteq \{N.$ *sets* $N =$ *sets* (*borel-of mtopology*) $\wedge$ N (*space* N) $\leq$ *ennreal* $r\}$
by (*simp add*: *assms*(*1*) *closedin-bounded-measures closure-of-minimal*)
have *tight*: *tight-on-set mtopology* (*LPm.mtopology closure-of* Γ)
unfolding *tight-on-set-def*
proof *safe*
fix *e* :: *real*
assume *e*: $0 < e$
then obtain K **where** K: *compactin mtopology* K $\bigwedge N.\ N \in \Gamma \Longrightarrow$ *measure* N (*space* $N - K$) $< e\ /\ 2$
by (*metis assms*(*3*) *tight-on-set-def zero-less-divide-iff zero-less-numeral*)
show $\exists K.$ *compactin mtopology* $K \wedge (\forall M{\in}$*LPm.mtopology closure-of* $\Gamma.$ *mea-*

sure M (*space M* − *K*) < *e*)
proof(*safe intro*!: *exI*[**where** *x*=*K*])
fix *N*
assume *N*:*N* ∈ *LPm.mtopology closure-of* Γ
then obtain *Nn* **where** *Nn*: ⋀*n*. *Nn n* ∈ Γ *limitin LPm.mtopology Nn N sequentially*
unfolding *LPm.closure-of-sequentially* **by** *auto*
with *N subst1* **interpret** *mweak-conv-fin M d Nn N sequentially*
using *closure-of-subset-topspace* **by**(*fastforce intro*!: *inP-mweak-conv-fin-all simp*: *closure-of-subset-topspace*)
have *space-Ni*:⋀*i*. *space* (*Nn i*) = *M*
by (*meson Nn*(*1*) *inP-D*(*3*) *subsetD subst1*)
have *openin mtopology* (*M* − *K*)
using *compactin-imp-closedin*[*OF Hausdorff-space-mtopology K*(*1*)] **by** *blast*
hence *ereal* (*measure N* (*M* − *K*)) ≤ *liminf* (λ*n*. *ereal* (*measure* (*Nn n*) (*M* −*K*)))
using *mweak-conv-eq3 converge-imp-mweak-conv*[*OF Nn*(*2*)] *Nn*(*1*) *subst1* **by** *blast*
also have ... ≤ *ereal* (*e* / *2*)
using *K*(*2*) *Nn*(*1*) *space-Ni*
by(*intro Liminf-le eventuallyI ereal-less-eq*(*3*)[*THEN iffD2*] *order.strict-implies-order*) *fastforce*+
also have ... < *ereal e*
using *e* **by** *auto*
finally show *measure N* (*space N* − *K*) < *e*
by(*auto simp*: *space-N*)
qed *fact*
qed(*use closure-of-subset-topspace*[*of LPm.mtopology* Γ] *inP-D* **in** *auto*)
show *?thesis*
unfolding *LPm.compactin-sequentially*
proof *safe*
fix *Ni* :: *nat* ⇒ *'a measure*
assume *Ni*: *range Ni* ⊆ *LPm.mtopology closure-of* Γ
then have *Ni2*: ⋀*i*. *finite-measure* (*Ni i*) **and** *Ni-le-r*: ⋀*i*. *Ni i* (*space* (*Ni i*)) ≤ *ennreal r*
and *sets-Ni*[*measurable-cong*]:⋀*i*. *sets* (*Ni i*) = *sets* (*borel-of mtopology*)
and *space-Ni*:⋀*i*. *space* (*Ni i*) = *M*
using *closure-of-subset-topspace*[*of LPm.mtopology* Γ] *inP-D subst2* **by** *fastforce*+
interpret *Ni*: *finite-measure Ni i* **for** *i*
by *fact*
have *metrizable-space Hilbert-cube-topology*
by(*auto simp*: *metrizable-space-product-topology metrizable-space-euclidean intro*!: *metrizable-space-subtopology*)
then obtain *dH* **where** *dH*: *Metric-space* (*UNIV* $\rightarrow_E$ {*0..1*}) *dH*
Metric-space.mtopology (*UNIV* $\rightarrow_E$ {*0..1*}) *dH* = *Hilbert-cube-topology*
by (*metis Metric-space.topspace-mtopology metrizable-space-def topspace-Hilbert-cube*)
then interpret *dH*: *Metric-space UNIV* $\rightarrow_E$ {*0..1*} *dH*
by *auto*

have *compact-dH*:*compact-space dH.mtopology*
unfolding *dH*(*2*) **by**(*auto simp*: *compact-space-def compactin-PiE*)
from *embedding-into-Hilbert-cube*[*OF metrizable-space-mtopology assms*(*2*)]
obtain *A* **where** *A*: *A* ⊆*topspace Hilbert-cube-topology*
mtopology homeomorphic-space subtopology Hilbert-cube-topology A
by *auto*
then obtain *T T-inv* **where** *T*: *continuous-map mtopology* (*subtopology Hilbert-cube-topology A*) *T*
continuous-map (*subtopology Hilbert-cube-topology A*) *mtopology T-inv*
⋀*x*. *x* ∈ *topspace* (*subtopology Hilbert-cube-topology A*)
⟹ *T* (*T-inv x*) = *x* ⋀*x*. *x* ∈ *M* ⟹ *T-inv* (*T x*) = *x*
unfolding *homeomorphic-space-def homeomorphic-maps-def* **by** *fastforce*
hence *injT*: *inj-on T M*
by(*intro inj-on-inverseI*)
have *T-cont*: *continuous-map mtopology dH.mtopology T*
by (*metis T*(*1*) *continuous-map-in-subtopology dH*(*2*))
from *continuous-map-measurable*[*OF this*]
have *T-meas*[*measurable*]: *T* ∈ *measurable* (*Ni n*) (*borel-of dH.mtopology*) **for** *n*
by(*auto simp*: *sets-Ni cong*: *measurable-cong-sets*)
define *νn* **where** *νn* ≡ (*λi*. *distr* (*Ni i*) (*borel-of dH.mtopology*) *T*)
have *sets-νn*: ⋀*n*. *sets* (*νn n*) = *sets* (*borel-of dH.mtopology*)
unfolding *νn-def* **by** *simp*
hence *space-νn*: ⋀*n*. *space* (*νn n*) = *UNIV* $\rightarrow_E$ {*0..1*}
by(*auto cong*: *sets-eq-imp-space-eq simp*: *space-borel-of*)
interpret *νn*: *finite-measure νn n* **for** *n*
by(*auto simp*: *νn-def space-borel-of PiE-eq-empty-iff intro*!: *Ni.finite-measure-distr*)
have *νn-le-r*: *νn n* (*space* (*νn n*)) ≤ *ennreal r* **for** *n*
by(*auto simp*: *νn-def emeasure-distr order.trans*[*OF emeasure-space Ni-le-r*[*of n*]])
have *measure-νn-compact*:*measure* (*νn n*) (*space* (*νn n*) − *T* ' *K*) = *measure* (*Ni n*) (*space* (*Ni n*) − *K*)
if *K*: *compactin mtopology K* **for** *K n*
proof −
have *compactin dH.mtopology* (*T* ' *K*)
using *T-cont image-compactin K* **by** *blast*
hence *T* ' *K* ∈ *sets* (*borel-of dH.mtopology*)
by(*auto intro*!: *borel-of-closed compactin-imp-closedin dH.Hausdorff-space-mtopology*)
hence *measure* (*νn n*) (*space* (*νn n*) − *T* ' *K*)
= *measure* (*Ni n*) (*T* −' (*space* (*νn n*) − *T* ' *K*) ∩ *space* (*Ni n*))
by(*simp add*: *νn-def measure-distr*)
also have ... = *measure* (*Ni n*) (*space* (*Ni n*) − *K*)
using *compactin-subset-topspace*[*OF K*] *T*(*4*) *Pi-mem*[*OF continuous-map-funspace*[*OF T*(*1*)]]
by(*auto intro*!: *arg-cong*[**where** *f*=*measure* (*Ni n*)] *simp*: *space-Ni subset-iff space-νn*) *metis*
finally show *?thesis* **.**
qed
define *HP* **where** *HP* ≡ {*N*. *sets N* = *sets* (*borel-of dH.mtopology*) ∧ *N* (*space*

N) ≤ *ennreal r*}
interpret *dHs*: *Levy-Prokhorov UNIV* $\to_E$ {*0..1*} *dH*
using *dH*(*1*) **by**(*auto simp*: *HP-def Levy-Prokhorov-def*)
have *HP*:*HP* ⊆ {*N*. *sets N* = *sets* (*borel-of dH.mtopology*) ∧ *finite-measure N*}
by(*auto simp*: *HP-def top.extremum-unique intro*!: *finite-measureI*)
have *νn-HP*:*range νn* ⊆ *HP*
by(*fastforce simp*: *HP-def sets-νn νn-le-r*)
then obtain *ν′ a* **where** *ν′*: *ν′* ∈ *HP strict-mono a limitin dHs.LPm.mtopology* (*νn* ∘ *a*) *ν′ sequentially*
using *dHs.mcompact-imp-LPmcompact*[*OF compact-dH*,*of r*]
unfolding *dHs.LPm.compactin-sequentially HP-def* **by** *meson*
hence *sets-ν′*[*measurable-cong*]: *sets ν′* = *sets* (*borel-of dH.mtopology*)
and *ν′-le-r*: *ν′* (*space ν′*) ≤ *ennreal r*
by(*auto simp*: *HP-def space-borel-of*)
have *space-ν′*: *space ν′* = *UNIV* $\to_E$ {*0..1*}
using *sets-eq-imp-space-eq*[*OF sets-ν′*] **by**(*simp add*: *space-borel-of*)
interpret *ν′*: *finite-measure ν′*
using *ν′-le-r* **by**(*auto intro*!: *finite-measureI simp*: *top-unique*)
interpret *wc*:*mweak-conv-fin UNIV* $\to_E$ {*0..1*} *dH νn* ∘ *a ν′ sequentially*
using *νn-HP HP* **by**(*fastforce intro*!: *dHs.inP-mweak-conv-fin-all ν′ dHs.inP-I*)
have *claim*: ∃ *E*⊆*A*. *E* ∈ *sets* (*borel-of dH.mtopology*) ∧ *measure ν′* (*space ν′* − *E*) = *0*
proof −
{
fix *n*
have ∃ *Kn*. *compactin mtopology Kn* ∧ (∀ *N*∈*LPm.mtopology closure-of* Γ. *measure N* (*space N* − *Kn*) < *1* / *Suc n*)
using *tight* **by**(*auto simp*: *tight-on-set-def*)
}
then obtain *Kn* **where** *Kn*: ⋀*n*. *compactin mtopology* (*Kn n*)
⋀*N n*. *N* ∈ *LPm.mtopology closure-of* Γ ⟹ *measure N* (*space N* − *Kn n*) < *1* / *Suc n*
by *metis*
have *TKn-compact*:⋀*n*. *compactin dH.mtopology* (*T* ‘ (*Kn n*))
by (*metis Kn*(*1*) *T-cont image-compactin*)
hence [*measurable*]:⋀*n*. *T* ‘ *Kn n* ∈ *sets* (*borel-of dH.mtopology*)
by(*auto intro*!: *borel-of-closed compactin-imp-closedin dH.Hausdorff-space-mtopology*)
have *T-img*: ⋀*n*. *T* ‘ (*Kn n*) ⊆ *A*
using *continuous-map-image-subset-topspace*[*OF T*(*1*)] *compactin-subset-topspace*[*OF Kn*(*1*)]
by *fastforce*
define *E* **where** *E* ≡ (⋃ *n*. *T* ‘ (*Kn n*))
have [*measurable*]: *E* ∈ *sets* (*borel-of dH.mtopology*)
by(*simp add*: *E-def*)
show *?thesis*
proof(*safe intro*!: *exI*[**where** *x*=*E*])
show *measure ν′* (*space ν′* − *E*) = *0*
proof(*rule antisym*[*OF field-le-epsilon*])
fix *e* :: *real*

assume *e*: *0* < *e*
then obtain *n0* **where** *n0*: *1* / (*Suc n0*) < *e*
using *nat-approx-posE* **by** *blast*
show *measure* ν′ (*space* ν′ − *E*) ≤ *0* + *e*
proof −
have *ereal* (*measure* ν′ (*space* ν′ − *E*)) ≤ *ereal* (*measure* ν′ (*space* ν′ − *T* ‘ (*Kn n0*)))
by(*auto intro*!: ν′.*finite-measure-mono simp*: *E-def*)
also have ... ≤ *liminf* (λ*n*. *ereal* (*measure* ((ν*n* ∘ *a*) *n*) (*space* ν′ − *T* ‘ (*Kn n0*))))
proof −
have *openin dH.mtopology* (*space* ν′ − *T* ‘ (*Kn n0*))
by (*metis TKn-compact compactin-imp-closedin dH.Hausdorff-space-mtopology dH.open-in-mspace openin-diff wc.space-N*)
with *wc.mweak-conv-eq3*[*THEN iffD1*,*OF dHs.converge-imp-mweak-conv*[*OF* ν′(*3*)]]
show *?thesis*
using ν*n-HP HP* **by**(*auto simp*: *dHs.inP-iff*)
qed
also have ... = *liminf* (λ*n*. *ereal* (*measure* ((ν*n* ∘ *a*) *n*) (*space* ((ν*n* ∘ *a*) *n*) − *T* ‘ (*Kn n0*))))
by(*auto simp*: *space*-ν*n space*-ν′)
also have ... = *liminf* (λ*n*. *ereal* (*measure* ((*Ni* ∘ *a*) *n*) (*space* ((*Ni* ∘ *a*) *n*) − *Kn n0*)))
by(*simp add*: *measure*-ν*n-compact*[*OF Kn*(*1*)])
also have ... ≤ *1* / (*Suc n0*)
using *Ni*
by(*intro Liminf-le eventuallyI ereal-less-eq*(*3*)[*THEN iffD2*] *order.strict-implies-order Kn*(*2*))
auto
also have ... < *ereal e*
using *n0* **by** *auto*
finally show *?thesis*
by *simp*
qed
qed *simp*
qed(*use E-def T-img* **in** *auto*)
qed
then obtain *E* **where** *E*[*measurable*]: *E* ⊆ *A*
E ∈ *sets* (*borel-of dH.mtopology*) *measure* ν′ (*space* ν′ − *E*) = *0*
by *blast*
have *measure*-ν′: *measure* ν′ (*B* ∩ *E*) = *measure* ν′ *B*
if *B*[*measurable*]: *B* ∈ *sets* (*borel-of dH.mtopology*) **for** *B*
proof(*rule antisym*)
have *measure* ν′ *B* = *measure* ν′ (*B* ∩ *E* ∪ *B* ∩ (*space* ν′ − *E*))
using *sets.sets-into-space*[*OF B*]
by(*auto intro*!: *arg-cong*[**where** *f*=*measure* ν′] *simp*: *space*-ν′ *space-borel-of*)
also have ... ≤ *measure* ν′ (*B* ∩ *E*) + *measure* ν′ (*B* ∩ (*space* ν′ − *E*))
by(*auto intro*!: *measure-Un-le*)

also have ... $\leq$ *measure* ν' $(B \cap E)$ + *measure* ν' ((*space* ν' $-$ E))
by(*auto intro*!: ν'.*finite-measure-mono*)
also have ... = *measure* ν' $(B \cap E)$
by(*simp add*: *E*)
finally show *measure* ν' *B* $\leq$ *measure* ν' $(B \cap E)$ **.**
qed(*auto intro*!: ν'.*finite-measure-mono*)
from *this*[*of space* ν'] *sets.sets-into-space*[*OF E*(*2*)]
have *measure-*ν'*E*:*measure* ν' *E* = *measure* ν' (*space* ν')
by(*auto simp*: *space-*ν' *borel-of-open space-borel-of inf.absorb-iff2*)
show $\exists N\ r.\ N \in$ *LPm.mtopology closure-of* Γ $\wedge$ *strict-mono r* $\wedge$ *limitin LPm.mtopology* (*Ni* $\circ$ *r*) *N sequentially*
proof $-$
define ν **where** $\nu \equiv$ *restrict-space* ν' *E*
interpret ν: *finite-measure* ν
by(*auto intro*!: *finite-measure-restrict-space* ν'.*finite-measure-axioms simp*: ν-*def*)
have *space-*ν:*space* ν = *E*
using *E*(*2*) ν-*def sets-*ν' *space-restrict-space2* **by** *blast*
have ν-*le-r*: ν (*space* ν) $\leq$ *ennreal r*
by(*simp add*: ν-*def emeasure-restrict-space order.trans*[*OF emeasure-space* ν'-*le-r*])
have *measure-*ν'*2*: *measure* ν' *B* = *measure* ν $(B \cap E)$
if *B*[*measurable*]: $B \in$ *sets* (*borel-of dH.mtopology*) **for** *B*
by(*auto simp*: ν-*def measure-restrict-space measure-*ν')
have *T-inv-measurable*[*measurable*]: *T-inv* $\in \nu \rightarrow_M$ *borel-of mtopology*
using *continuous-map-measurable*[*OF continuous-map-from-subtopology-mono*[*OF T*(*2*) *E*(*1*)]]
by(*auto simp*: ν-*def borel-of-subtopology dH*
cong: *sets-restrict-space-cong*[*OF sets-*ν'] *measurable-cong-sets*)
define *N* **where** $N \equiv$ *distr* ν (*borel-of mtopology*) *T-inv*
have *N-inP*:$N \in \mathcal{P}$
using *Ni2*[*of 0*,*simplified subprob-space-def subprob-space-axioms-def*]
by(*auto simp*: $\mathcal{P}$-*def N-def space-Ni emeasure-distr order.trans*[*OF emeasure-space* ν-*le-r*] ν.*finite-measure-distr*)
then interpret *wcN*:*mweak-conv-fin M d Ni* $\circ$ *a N sequentially*
using *subset-trans*[*OF Ni closure-of-subset-topspace*] **by**(*auto intro*!: *inP-mweak-conv-fin-all*)

show $\exists N\ r.\ N \in$ *LPm.mtopology closure-of* Γ $\wedge$ *strict-mono r* $\wedge$ *limitin LPm.mtopology* (*Ni* $\circ$ *r*) *N sequentially*
proof(*safe intro*!: *exI*[**where** *x*=*N*] *exI*[**where** *x*=*a*])
show *limit*: *limitin LPm.mtopology* (*Ni* $\circ$ *a*) *N sequentially*
proof(*rule mweak-conv-imp-converge*)
show *mweak-conv-seq* (*Ni* $\circ$ *a*) *N*
unfolding *wcN.mweak-conv-eq2*
proof *safe*
have [*measurable*]:*UNIV* $\rightarrow_E$ {*0..1*} $\in$ *sets* (*borel-of dH.mtopology*)
by(*auto simp*: *borel-of-open*)
have *1*:*measure* ((*Ni* $\circ$ *a*) *n*) *M* = *measure* ((νn $\circ$ *a*) *n*) (*UNIV* $\rightarrow_E$ {*0..1*}) **for** *n*

using *continuous-map-funspace*[*OF T*(*1*)]
by(*auto simp*: *νn-def measure-distr space-Ni intro*!: *arg-cong*[**where** *f*=*measure* (*Ni* (*a n*))])
have *2*: *measure N M* = *measure ν′* (*space ν′*)
proof −
have [*measurable*]: *M* ∈ *sets* (*borel-of mtopology*)
by(*auto intro*!: *borel-of-open*)
have *measure N M* = *measure ν* (*T-inv* −‘ *M* ∩ *space ν*)
by(*auto simp*: *N-def intro*!: *measure-distr*)
also have ... = *measure ν* (*space ν* ∩ *E*)
using *measurable-space*[*OF T-inv-measurable*]
by(*auto intro*!: *arg-cong*[**where** *f*=*measure ν*] *simp*: *space-borel-of space-ν*)
also have ... = *measure ν′* (*space ν*)
by(*rule measure-ν′2*[*symmetric*]) (*simp add*: *space-ν*)
also have ... = *measure ν′* (*space ν′*)
by(*simp add*: *measure-ν′E space-ν*)
finally show *?thesis* **.**
qed
show (*λn. measure* ((*Ni* ∘ *a*) *n*) *M*) ⟶ *measure N M*
unfolding *1 2* **using** *HP νn-HP wc.mweak-conv-eq2*[*THEN iffD1*,*OF dHs.converge-imp-mweak-conv*[*OF ν′*(*3*)]]
by(*auto simp*: *space-ν′ dHs.inP-iff*)
next
fix *C*
assume *C*: *closedin mtopology C*
hence [*measurable*]: *C* ∈ *sets* (*borel-of mtopology*)
by(*auto intro*!: *borel-of-closed*)
have *closedin* (*subtopology dH.mtopology A*) (*T* ‘ *C*)
proof −
have *T* ‘ *C* = {*x* ∈ *topspace* (*subtopology Hilbert-cube-topology A*). *T-inv x* ∈ *C*}
using *closedin-subset*[*OF C*] *T*(*3*,*4*) *continuous-map-funspace*[*OF T*(*1*)] *continuous-map-funspace*[*OF T*(*2*)]
by (*auto simp*: *rev-image-eqI*)
also note *closedin-continuous-map-preimage*[*OF T*(*2*) *C*]
finally show *?thesis*
by(*simp add*: *dH*)
qed
then obtain *K* **where** *K*: *closedin dH.mtopology K T* ‘ *C* = *K* ∩ *A*
by (*meson closedin-subtopology*)
hence [*measurable*]: *K* ∈ *sets* (*borel-of dH.mtopology*)
by(*simp add*: *borel-of-closed*)
have *C-eq*:*C* = *T* −‘ *K* ∩ *M*
proof −
have *C* = (*T* −‘ *T* ‘ *C*) ∩ *M*
using *closedin-subset*[*OF C*] *injT* **by**(*auto dest*: *inj-onD*)
also have ... = (*T* −‘ (*K* ∩ *A*)) ∩ *M*
by(*simp only*: *K*(*2*))

also have ... = $T -`K \cap M$
using $A(1)$ *continuous-map-funspace*[*OF* $T(1)$] **by** *auto*
finally show *?thesis* **.**
qed
hence *1*:*measure* ((*Ni* ∘ *a*) *n*) *C* = *measure* ((*νn* ∘ *a*) *n*) *K* **for** *n*
by(*auto simp*: *νn-def measure-distr space-Ni*)
have *limsup* (λ*n*. *ereal* (*measure* ((*Ni* ∘ *a*) *n*) *C*)) = *limsup* (λ*n*. *ereal* (*measure* ((*νn* ∘ *a*) *n*) *K*))
unfolding *1* **by** *simp*
also have ... ≤ *ereal* (*measure* *ν'* *K*)
using *νn-HP HP wc.mweak-conv-eq2*[*THEN iffD1*,*OF dHs.converge-imp-mweak-conv*[*OF* *ν'*(*3*)]] *K*(*1*) *dHs.inP-iff* **by** *auto*
also have ... = *ereal* (*measure* *ν* (*K* ∩ *E*))
by(*simp add*: *measure-ν'2*)
also have ... = *ereal* (*measure* *ν* (*T-inv* −` *C* ∩ *space* *ν*))
using *measurable-space*[*OF T-inv-measurable*] *K*(*2*) *E*(*1*) *closedin-subset*[*OF* *K*(*1*)] *A*(*1*) *T*(*3*,*4*)
by(*fastforce intro*!: *arg-cong*[**where** *f*=*measure* *ν*] *simp*: *space-ν C-eq space-borel-of subsetD*)
also have ... = *ereal* (*measure* *N* *C*)
by(*auto simp*: *N-def measure-distr*)
finally show *limsup* (λ*n*. *ereal* (*measure* ((*Ni* ∘ *a*) *n*) *C*)) ≤ *ereal* (*measure* *N* *C*) **.**
qed
qed(*use N-inP Ni assms closure-of-subset-topspace*[*of LPm.mtopology* Γ] **in** *auto*)
have *range* (*Ni* ∘ *a*) ⊆ *LPm.mtopology closure-of* Γ
using *Ni* **by** *auto*
thus *N* ∈ *LPm.mtopology closure-of* Γ
using *limit LPm.metric-closedin-iff-sequentially-closed*[*THEN iffD1*,*OF closedin-closure-of*[*of* - Γ]]
by *blast*
qed *fact*
qed
qed(*use assms*(*1*) *closedin-subset*[*OF closedin-closure-of*[*of LPm.mtopology*]] **in** *auto*)
qed

corollary *Prokhorov-theorem-LP*:
assumes Γ ⊆ {*N*. *sets* *N* = *sets* (*borel-of mtopology*) ∧ *emeasure* *N* (*space* *N*) ≤ *ennreal* *r*}
and *separable-space mtopology mcomplete*
shows *compactin LPm.mtopology* (*LPm.mtopology closure-of* Γ) ⟷ *tight-on-set mtopology* Γ
proof −
have Γ ⊆ 𝒫
using *assms*(*1*) **by**(*auto intro*!: *finite-measureI inP-I simp*: *top.extremum-unique*)
thus *?thesis*
using *assms* **by**(*auto simp*: *relatively-compact-imp-tight-LP tight-imp-relatively-compact-LP*)

qed

5.2 Completeness of the Lévy-Prokhorov Metric

lemma *mcomplete-tight-on-set*:
assumes $\Gamma \subseteq \mathcal{P}$ *mcomplete*
and $\bigwedge e\ f.\ e > 0 \Longrightarrow f > 0$
$\Longrightarrow \exists\, an\ n.\ an$ ‘ $\{..n::nat\} \subseteq M \wedge (\forall N\in\Gamma.\ measure\ N\ (M - (\bigcup i\leq n.\ mball\ (an\ i)\ f)) \leq e)$
shows *tight-on-set mtopology* Γ
unfolding *tight-on-set-def*
proof *safe*
fix *e* :: *real*
assume *e*: $0 < e$
then have $\exists\, an\ n.\ an$ ‘ $\{..n::nat\} \subseteq M \wedge$
$(\forall N\in\Gamma.\ measure\ N\ (M - (\bigcup i\leq n.\ mball\ (an\ i)\ (1\ /\ (1 + real\ m)))) \leq e\ /\ 2 * (1\ /\ 2)\ \hat{}\ Suc\ m)$ **for** *m*
using *assms*(*3*)[*of e* / *2* ∗ (*1* / *2*) ^ *Suc m 1* / (*1* + *real m*)] **by** *fastforce*
then obtain *anm nm* **where** *anm*: $\bigwedge m.\ anm\ m$ ‘ $\{..nm\ m::nat\} \subseteq M$
$\bigwedge m\ N.\ N \in \Gamma \Longrightarrow measure\ N\ (M - (\bigcup i\leq nm\ m.\ mball\ (anm\ m\ i)\ (1\ /\ (1 + real\ m)))) \leq e/\ 2 * (1\ /\ 2)\hat{}Suc\ m$
by *metis*
define *K* **where** $K \equiv (\bigcap m.\ (\bigcup i\leq nm\ m.\ mcball\ (anm\ m\ i)\ (1\ /\ (1 + real\ m))))$
have *K-closed*: *closedin mtopology K*
by(*auto simp*: *K-def intro*!: *closedin-Union*)
show $\exists K.\ compactin\ mtopology\ K \wedge (\forall M\in\Gamma.\ measure\ M\ (space\ M - K) < e)$
proof(*safe intro*!: *exI*[**where** *x*=*K*])
have *mtotally-bounded K*
unfolding *mtotally-bounded-def2*
proof *safe*
fix ε :: *real*
assume ε: $0 < \varepsilon$
then obtain *m* **where** *m*: $1\ /\ (1 + real\ m) < \varepsilon$
using *nat-approx-posE* **by** *auto*
show $\exists Ka.\ finite\ Ka \wedge Ka \subseteq M \wedge K \subseteq (\bigcup x\in Ka.\ mball\ x\ \varepsilon)$
proof(*safe intro*!: *exI*[**where** *x*=*anm m* ‘ {..*nm m*}])
fix *x*
assume $x \in K$
then have $x \in (\bigcup i\leq nm\ m.\ mcball\ (anm\ m\ i)\ (1\ /\ (1 + real\ m)))$
by(*auto simp*: *K-def*)
also have ... $\subseteq (\bigcup i\leq nm\ m.\ mball\ (anm\ m\ i)\ \varepsilon)$
by(*rule UN-mono*) (*use m* **in** *auto*)
finally show $x \in (\bigcup x\in anm\ m$ ‘ $\{..nm\ m\}.\ mball\ x\ \varepsilon)$
by *auto*
qed(*use anm* **in** *auto*)
qed
thus *compactin mtopology K*
by(*simp add*: *mtotally-bounded-eq-compact-closedin*[*OF assms*(*2*) *K-closed*])
next

fix N
assume N:N ∈ Γ
then interpret N: finite-measure N
using assms(1) inP-D(1) **by** auto
have [measurable]: M ∈ sets N ⋀a b. mcball a b ∈ sets N
using N inP-D(2) assms(1) **by**(auto intro!: borel-of-closed)
have [measurable]: ⋀a b. mball a b ∈ sets N
using N inP-D(2) assms(1) **by**(auto intro!: borel-of-open)
have [simp]: summable (λm. measure N (M − (⋃i≤nm m. mball (anm m i) (1 / (1 + real m)))))
using anm(2)[OF N]
by(auto intro!: summable-comparison-test-ev[**where** g=λn. e / 2 * (1 / 2) ^ Suc n
and f=λm. measure N (M − (⋃i≤nm m. mball (anm m i) (1 / (1 + real m))))] eventuallyI)
show measure N (space N − K) < e
proof −
have measure N (space N − K) = measure N (M − K)
using N assms(1) inP-D(3) **by** auto
also have ... = measure N (⋃m. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
by(auto simp: K-def)
also have ... ≤ (∑m. e / 2 * (1 / 2) ^ Suc m)
proof −
have (λk. measure N (⋃m≤k. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m)))))
⟶ measure N (⋃i. ⋃m≤i. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
by(rule N.finite-Lim-measure-incseq) (auto intro!: incseq-SucI)
moreover have (⋃i. ⋃m≤i. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
= (⋃m. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
by blast
ultimately have 1:(λk. measure N (⋃m≤k. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m)))))
⟶ measure N (⋃m. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
by simp
show ?thesis
proof(safe intro!: Lim-bounded[OF 1])
fix n
show measure N (⋃m≤n. M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m))))
≤ (∑m. e / 2 * (1 / 2) ^ Suc m) (**is** ?lhs ≤ ?rhs)
proof −
have ?lhs ≤ (∑m≤n. measure N (M − (⋃i≤nm m. mcball (anm m i) (1 / (1 + real m)))))
by(rule N.finite-measure-subadditive-finite) auto

```
          also have ... ≤ (∑m≤n. measure N (M − (⋃i≤nm m. mball (anm m
i) (1 / (1 + real m)))))
            by(rule sum-mono) (auto intro!: N.finite-measure-mono)
          also have ... ≤ (∑m. measure N (M − (⋃i≤nm m. mball (anm m i)
(1 / (1 + real m)))))
            by(rule sum-le-suminf) auto
          also have ... ≤ ?rhs
            by(rule suminf-le) (use anm(2)[OF N] in auto)
          finally show ?thesis .
        qed
      qed
    qed
    also have ... = e / 2 * (∑m. (1 / 2) ^ Suc m)
      by(rule suminf-mult) auto
    also have ... = e / 2
      using power-half-series sums-unique by fastforce
    also have ... < e
      using e by simp
    finally show ?thesis .
  qed
 qed
qed(use assms(1) inP-D in auto)

lemma mcomplete-LPmcomplete:
 assumes mcomplete separable-space mtopology
 shows LPm.mcomplete
proof −
 consider M = {} | M ≠ {}
  by blast
 then show ?thesis
 proof cases
  case 1
  from M-empty-P[OF this]
  have 𝒫 = {} ∨ 𝒫 = {count-space {}} .
  then show ?thesis
  using LPm.compact-space-eq-Bolzano-Weierstrass LPm.compact-space-imp-mcomplete
finite-subset
   by fastforce
 next
  case M-ne:2
  show ?thesis
   unfolding LPm.mcomplete-def
  proof safe
   fix Ni
   assume cauchy: LPm.MCauchy Ni
   hence range-Ni: range Ni ⊆ 𝒫
    by(auto simp: LPm.MCauchy-def)
   hence range-Ni2: range Ni ⊆ LPm.mtopology closure-of (range Ni)
    by (simp add: closure-of-subset)
```

```
    have Ni-inP: ⋀i. Ni i ∈ 𝒫
      using cauchy by(auto simp: LPm.MCauchy-def)
    hence ⋀n. finite-measure (Ni n)
      and sets-Ni[measurable-cong]:⋀n. sets (Ni n) = sets (borel-of mtopology)
      and space-Ni: ⋀n. space (Ni n) = M
      by(auto dest: inP-D)
    then interpret Ni: finite-measure Ni n for n
      by simp
    have ∃r≥0. ∀i. Ni i (space (Ni i)) ≤ ennreal r
    proof −
      obtain N where N: ⋀n m. n ≥ N ⟹ m ≥ N ⟹ LPm (Ni n) (Ni m)
< 1
        using LPm.MCauchy-def cauchy zero-less-one by blast
      define r where r = max (Max ((λi. measure (Ni i) (space (Ni i))) ‘ {..N}))
(measure (Ni N) (space (Ni N)) + 1)
      show ?thesis
      proof(safe intro!: exI[where x=r])
        fix i
        consider i ≤ N | N ≤ i
          by fastforce
        then show Ni i (space (Ni i)) ≤ ennreal r
        proof cases
          assume i ≤ N
          then have measure (Ni i) (space (Ni i)) ≤ r
            by(auto simp: r-def intro!: max.coboundedI1)
          thus ?thesis
            by (simp add: measure-def enn2real-le)
        next
          assume i:i ≥ N
          have measure (Ni i) (space (Ni i)) ≤ r
          proof −
            have measure (Ni i) M ≤ measure (Ni N) (⋃a∈M. mball a 1) + 1
          using range-Ni by(auto intro!: LPm-less-then[of Ni N] N i borel-of-open)
            also have ... ≤ measure (Ni N) (space (Ni N)) + 1
              using Ni.bounded-measure by auto
            also have ... ≤ r
              by(auto simp: r-def)
            finally show ?thesis
              by(simp add: space-Ni)
          qed
          thus ?thesis
            by (simp add: Ni.emeasure-eq-measure ennreal-leI)
        qed
      qed(auto simp: r-def intro!: max.coboundedI2)
    qed
    then obtain r where r-nonneg: r ≥ 0 and r-bounded:⋀i. Ni i (space (Ni
i)) ≤ ennreal r
      by blast
    with sets-Ni have range-Ni':
```

range Ni ⊆ {*N*. *sets N* = *sets* (*borel-of mtopology*) ∧ *emeasure N* (*space N*) ≤ *ennreal r*}
by *blast*
have *M-meas*[*measurable*]: *M* ∈ *sets* (*borel-of mtopology*)
by(*simp add*: *borel-of-open*)
have *mball-meas*[*measurable*]: *mball a e* ∈ *sets* (*borel-of mtopology*) **for** *a e*
by(*auto intro*!: *borel-of-open*)
have *Ni-Cauchy*: ⋀*e*. *e* > *0* ⟹ ∃ *n0*. ∀ *n n'*. *n0* ≤ *n* ⟶ *n0* ≤ *n'* ⟶ *LPm* (*Ni n*) (*Ni n'*) < *e*
using *cauchy* **by**(*auto simp*: *LPm.MCauchy-def*)
have *tight-on-set mtopology* (*range Ni*)
proof(*rule mcomplete-tight-on-set*[*OF range-Ni assms*(*1*)])
fix *e f* :: *real*
assume *e*: *e* > *0* **and** *f*: *f* > *0*
with *Ni-Cauchy*[*of min e f* / *2*] **obtain** *n0* **where** *n0*:
⋀*n m*. *n0* ≤ *n* ⟹ *n0* ≤ *m* ⟹ *LPm* (*Ni n*) (*Ni m*) < *min e f* / *2*
by *fastforce*
obtain *D* **where** *D*: *mdense D countable D*
using *assms*(*2*) *separable-space-def2* **by** *blast*
then obtain *an* **where** *an*: ⋀*n*::*nat*. *an n* ∈ *D range an* = *D*
by (*metis M-ne mdense-empty-iff rangeI uncountable-def*)
have ∃ *n1*. ∀ *i*≤*n0*. *measure* (*Ni i*) (*M* − (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))) ≤ *min e f* / *2*
proof −
have ∃ *n1*. *measure* (*Ni i*) (*M* − (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))) ≤ *min e f* / *2* **for** *i*
proof −
have (*λn1*. *measure* (*Ni i*) (*M* − (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*)))) ⟶ *0*
proof −
have *1*: (*λn1*. *measure* (*Ni i*) (*M* − (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))))
= (*λn1*. *measure* (*Ni i*) *M* − *measure* (*Ni i*) ((⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))))
using *Ni.finite-measure-compl* **by**(*auto simp*: *space-Ni*)
have (*λn1*. *measure* (*Ni i*) ((⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*)))) ⟶ *measure* (*Ni i*) *M*
proof −
have (*λn1*. *measure* (*Ni i*) ((⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))))
⟶ *measure* (*Ni i*) (⋃ *n1*. (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*)))
by(*intro Ni.finite-Lim-measure-incseq incseq-SucI UN-mono*) *auto*
moreover have (⋃ *n1*. (⋃ *i*≤*n1*. *mball* (*an i*) (*f* / *2*))) = *M*
using *mdense-balls-cover*[*OF D*(*1*)[*simplified an*(*2*)[*symmetric*]],*of f* / *2*] *f* **by** *auto*
ultimately show *?thesis* **by** *argo*
qed
from *tendsto-diff*[*OF tendsto-const*[**where** *k*=*measure* (*Ni i*) *M*] *this*] **show** *?thesis*
unfolding *1* **by** *simp*
qed

```
          thus ?thesis
        by (meson e f LIMSEQ-le-const half-gt-zero less-eq-real-def linorder-not-less
min-less-iff-conj)
        qed
        then obtain ni where ni: ⋀i. measure (Ni i) (M − (⋃i≤ni i. mball (an
i) (f / 2))) ≤ min e f / 2
          by metis
        define n1 where n1 ≡ Max (ni ‘ {..n0})
        show ?thesis
        proof(safe intro!: exI[where x=n1])
          fix i
          assume i: i ≤ n0
          then have nii:ni i ≤ n1
            by (simp add: n1-def)
          show measure (Ni i) (M − (⋃i≤n1. mball (an i) (f / 2))) ≤ min e f
/ 2
          proof −
            have measure (Ni i) (M − (⋃i≤n1. mball (an i) (f / 2)))
                  ≤ measure (Ni i) (M − (⋃i≤ni i. mball (an i) (f / 2)))
              using nii by(fastforce intro!: Ni.finite-measure-mono)
            also have ... ≤ min e f / 2
              by fact
            finally show ?thesis .
          qed
        qed
      qed
      then obtain n1 where n1:
        ⋀i. i ≤ n0 ⟹ measure (Ni i) (M − (⋃i≤n1. mball (an i) (f / 2))) ≤
e / 2
        ⋀i. i ≤ n0 ⟹ measure (Ni i) (M − (⋃i≤n1. mball (an i) (f / 2))) ≤ f
/ 2
        by auto
        show ∃an n. an ‘ {..n::nat} ⊆ M ∧ (∀N∈range Ni. measure N (M −
(⋃i≤n. mball (an i) f)) ≤ e)
      proof(safe intro!: exI[where x=an] exI[where x=n1])
        fix n
        consider n ≤ n0 | n0 ≤ n
          by linarith
        then show measure (Ni n) (M − (⋃i≤n1. mball (an i) f)) ≤ e
        proof cases
          case 1
          have measure (Ni n) (M − (⋃i≤n1. mball (an i) f))
                ≤ measure (Ni n) (M − (⋃i≤n1. mball (an i) (f / 2)))
            using f by(fastforce intro!: Ni.finite-measure-mono)
          also have ... ≤ e
            using n1[OF 1] e by linarith
          finally show ?thesis .
        next
          case 2
```

```
          have measure (Ni n) (M − (⋃i≤n1. mball (an i) f))
              ≤ measure (Ni n0) (⋃a∈M − (⋃i≤n1. mball (an i) f). mball a
(min e f / 2)) + min e f / 2
            by(intro LPm-less-then(2) n0 2 Ni-inP) auto
          also have ... ≤ measure (Ni n0) (M − (⋃i≤n1. mball (an i) (f / 2)))
+ min e f / 2
          proof −
            have (⋃a∈M − (⋃i≤n1. mball (an i) f). mball a (min e f / 2))
                  ⊆ M − (⋃i≤n1. mball (an i) (f / 2))
            proof safe
              fix x a i
              assume x: x ∈ mball a (min e f / 2)  x ∈ mball (an i) (f / 2)
                and a:a ∈ M a ∉ (⋃i≤n1. mball (an i) f) and i:i ≤ n1
              hence d (an i) x < f / 2 d x a < f / 2
                by(auto simp: commute)
              hence d (an i) a < f
                using triangle[of an i x a] a(1) x(2) by auto
              with a(2) i
              show False
                using a(1) atMost-iff image-eqI x(2) by auto
            qed simp
            thus ?thesis
              by(auto intro!: Ni.finite-measure-mono)
          qed
          also have ... ≤ e
            using n1(1)[OF order.refl] by linarith
          finally show ?thesis .
        qed
      qed(use an dense-in-subset[OF D(1)] in auto)
    qed
    from tight-imp-relatively-compact-LP[OF range-Ni' assms(2) this] range-Ni2
   obtain l N where strict-mono l limitin LPm.mtopology (Ni ∘ l) N sequentially

      unfolding LPm.compactin-sequentially by blast
    from LPm.MCauchy-convergent-subsequence[OF cauchy this]
    show ∃N. limitin LPm.mtopology Ni N sequentially
      by blast
  qed
 qed
qed
```

5.3 Equivalence of Separability, Completeness, and Compactness

lemma *return-inP*[*simp*]:*return* (*borel-of mtopology*) $x \in \mathcal{P}$
by (*metis emeasure-empty ennreal-top-neq-zero finite-measureI inP-I infinity-ennreal-def sets-return space-return subprob-space.axioms*(*1*) *subprob-space-return-ne*)

lemma *LPm-return-eq*:

assumes *x* ∈ *M* *y* ∈ *M*
shows *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) = *min 1* (*d x y*)
proof(*rule antisym*[*OF min.boundedI*])
show *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) ≤ *d x y*
proof(*rule field-le-epsilon*)
fix *e* :: *real*
assume *e*: *e* > *0*
show *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) ≤ *d x y* + *e*
proof(*rule LPm-imp-le*)
fix *B*
assume *B*[*measurable*]: *B* ∈ *sets* (*borel-of mtopology*)
have *x* ∈ *B* ⟹ *y* ∈ (⋃ *a*∈*B*. *mball a* (*d x y* + *e*))
using *e assms* **by** *auto*
thus *measure* (*return* (*borel-of mtopology*) *x*) *B*
≤ *measure* (*return* (*borel-of mtopology*) *y*) (⋃ *a*∈*B*. *mball a* (*d x y* + *e*)) + (*d x y* + *e*)
using *e* **by**(*simp add*: *measure-return indicator-def*)
next
fix *B*
assume *B*[*measurable*]: *B* ∈ *sets* (*borel-of mtopology*)
have *y* ∈ *B* ⟹ *x* ∈ (⋃ *a*∈*B*. *mball a* (*d x y* + *e*))
using *e assms* **by** (*auto simp*: *commute*)
thus *measure* (*return* (*borel-of mtopology*) *y*) *B*
≤ *measure* (*return* (*borel-of mtopology*) *x*) (⋃ *a*∈*B*. *mball a* (*d x y* + *e*)) + (*d x y* + *e*)
using *e* **by**(*simp add*: *measure-return indicator-def*)
qed (*simp add*: *add.commute add-pos-nonneg e*)
qed
next
consider *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) < *1*
| *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) ≥ *1*
by *linarith*
then show *min 1* (*d x y*)≤ *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*)
proof *cases*
case *1*
have *2*:*d x y* < *a* **if** *a*: *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) < *a*
a < *1* **for** *a*
proof −
have [*measurable*]: {*x*} ∈ *sets* (*borel-of mtopology*)
using *assms* **by**(*auto simp add*: *closedin-t1-singleton t1-space-mtopology intro*!: *borel-of-closed*)
have *measure* (*return* (*borel-of mtopology*) *x*) {*x*}
≤ *measure* (*return* (*borel-of mtopology*) *y*) (⋃ *b*∈{*x*}. *mball b a*) + *a*

using *assms subprob-space.subprob-emeasure-le-1*[*OF subprob-space-return-ne*[*of borel-of mtopology*]]
by(*intro LPm-less-then*(*1*)[**where** *A*={*x*},*OF* - - *a*(*1*)])
(*auto simp*: *space-borel-of space-scale-measure*)
thus *?thesis*
using *assms a*(*2*) *linorder-not-less* **by**(*fastforce simp*: *measure-return indicator-def*)
qed
have *d x y* < *a* **if** *a*: *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) < *a* **for** *a*
proof(*cases a* < *1*)
assume *r1*:~ *a* < *1*
obtain *k* **where** *k*:*LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) < *k k* < *1*
using *dense 1* **by** *blast*
show *?thesis*
using *2*[*OF k*] *k*(*2*) *r1* **by** *linarith*
qed(*use 2 a* **in** *auto*)
thus *?thesis*
by *force*
qed *simp*
next
show *LPm* (*return* (*borel-of mtopology*) *x*) (*return* (*borel-of mtopology*) *y*) ≤ *1*
by(*rule order.trans*[*OF LPm-le-max-measure*])
(*metis assms*(*1*) *assms*(*2*) *indicator-simps*(*1*) *max.idem measure-return nle-le sets.top space-borel-of space-return topspace-mtopology*)
qed

corollary *LPm-return-eq-capped-dist*:
assumes *x* ∈ *M y* ∈ *M*
shows *LPm* (*return* (*borel-of mtopology*) *x*)(*return* (*borel-of mtopology*) *y*) = *capped-dist 1 x y*
by(*simp add*: *capped-dist-def assms LPm-return-eq*)

corollary *MCauchy-iff-MCauchy-return*:
assumes *range xn* ⊆ *M*
shows *MCauchy xn* ⟷ *LPm.MCauchy* (λ*n*. *return* (*borel-of mtopology*) (*xn n*))
proof −
interpret *c*: *Metric-space M capped-dist 1*
using *capped-dist* **by** *blast*
show *?thesis*
using *range-subsetD*[*OF assms*(*1*)]
by(*auto simp*: *MCauchy-capped-metric*[*of 1*,*symmetric*] *c.MCauchy-def LPm.MCauchy-def LPm-return-eq-capped-dist*)
qed

lemma *conv-conv-return*:
assumes *limitin mtopology xn x sequentially*
shows *limitin LPm.mtopology* (λ*n*. *return* (*borel-of mtopology*) (*xn n*)) (*return*

(*borel-of mtopology*) *x*) *sequentially*
proof −
  interpret *c*: *Metric-space M capped-dist 1*
    using *capped-dist* **by** *blast*
  have *clim*:*limitin c.mtopology xn x sequentially*
    using *assms* **by** (*simp add*: *mtopology-capped-metric*)
  show *?thesis*
    using *LPm-return-eq-capped-dist clim*
    by(*fastforce simp*: *c.limit-metric-sequentially LPm.limit-metric-sequentially*)
qed

lemma *conv-iff-conv-return*:
  assumes *range xn* ⊆ *M x* ∈ *M*
  shows *limitin mtopology xn x sequentially*
        ⟷ *limitin LPm.mtopology* (*λn. return* (*borel-of mtopology*) (*xn n*))
                              (*return* (*borel-of mtopology*) *x*) *sequentially*
proof −
  have *xn*: ⋀*n. xn n* ∈ *M*
    using *assms* **by** *auto*
  interpret *c*: *Metric-space M capped-dist 1*
    using *capped-dist* **by** *blast*
  have *limitin mtopology xn x sequentially* ⟷ *limitin c.mtopology xn x sequentially*
    by (*simp add*: *mtopology-capped-metric*)
  also have ...
    ⟷ *limitin LPm.mtopology* (*λn. return* (*borel-of mtopology*) (*xn n*)) (*return*
(*borel-of mtopology*) *x*) *sequentially*
    using *xn assms* **by**(*auto simp*: *c.limit-metric-sequentially LPm.limit-metric-sequentially*
LPm-return-eq-capped-dist)
  finally show *?thesis* **.**
qed

lemma *continuous-map-return*: *continuous-map mtopology LPm.mtopology* (*λx.
return* (*borel-of mtopology*) *x*)
  by(*auto simp*: *continuous-map-iff-limit-seq*[*OF first-countable-mtopology*] *conv-conv-return*)

lemma *homeomorphic-map-return*:
  homeomorphic-map mtopology
                  (*subtopology LPm.mtopology* ((*λx. return* (*borel-of mtopology*) *x*) ‘
M))
                  (*λx. return* (*borel-of mtopology*) *x*)
proof(*rule homeomorphic-maps-imp-map*)
  define *inv* **where** *inv* ≡ (*λN. THE x. x* ∈ *M* ∧ *N* = *return* (*borel-of mtopology*)
x)
  have *inv-eq*: *inv* (*return* (*borel-of mtopology*) *x*) = *x* **if** *x*: *x* ∈ *M* **for** *x*
  proof −
    have *inv* (*return* (*borel-of mtopology*) *x*) ∈ *M* ∧ *return* (*borel-of mtopology*) *x*
          = *return* (*borel-of mtopology*) (*inv* (*return* (*borel-of mtopology*) *x*))
      unfolding *inv-def*
    proof(*rule theI*)

```
      fix y
      assume y ∈ M ∧ return (borel-of mtopology) x = return (borel-of mtopology)
y
      then show y = x
        using LPm-return-eq[OF x,of y] x
        by (auto intro!: zero[THEN iffD1] simp: commute simp del: zero)
    qed(use x in auto)
    thus ?thesis
      by (metis LPm-return-eq-capped-dist Metric-space.zero capped-dist x)
  qed
  interpret s: Submetric 𝒫 LPm (λx. return (borel-of mtopology) x) ‘ M
    by standard auto
  have continuous-map mtopology s.sub.mtopology (λx. return (borel-of mtopology)
x)
    using continuous-map-return
   by (simp add: LPm.Metric-space-axioms metric-continuous-map s.sub.Metric-space-axioms)
  moreover have continuous-map s.sub.mtopology mtopology inv
    unfolding continuous-map-iff-limit-seq[OF s.sub.first-countable-mtopology]
  proof safe
    fix Ni N
    assume h:limitin s.sub.mtopology Ni N sequentially
    then obtain x where x: x ∈ M N = return (borel-of mtopology) x
      using s.sub.limit-metric-sequentially by auto
    interpret c: Metric-space M capped-dist 1
      using capped-dist by blast
    show limitin mtopology (λn. inv (Ni n)) (inv N) sequentially
     unfolding c.limit-metric-sequentially mtopology-capped-metric[of 1,symmetric]
    proof safe
      fix e :: real
      assume e > 0
      then obtain n0 where n0:
        ⋀n. n ≥ n0 ⟹ Ni n ∈ (λx. return (borel-of mtopology) x) ‘ M
        ⋀n. n ≥ n0 ⟹ LPm (Ni n) N < e
        by (metis h s.sub.limit-metric-sequentially)
      then obtain xn where xn: ⋀n. n ≥ n0 ⟹ xn n ∈ M
        ⋀n. n ≥ n0 ⟹ Ni n = return (borel-of mtopology) (xn n)
        unfolding image-def by simp metis
     thus ∃Na. ∀n≥Na. inv (Ni n) ∈ M ∧ capped-dist 1 (inv (Ni n)) (inv N) < e
      using n0 by(auto intro!: exI[where x=n0] simp: inv-eq x LPm-return-eq-capped-dist)
    qed(simp add: inv-eq x)
  qed
  moreover have ∀x∈topspace mtopology. inv (return (borel-of mtopology) x) = x
    ∀y∈topspace s.sub.mtopology. return (borel-of mtopology) (inv y) = y
    by(auto simp: inv-eq)
   ultimately show homeomorphic-maps mtopology (subtopology LPm.mtopology
((λx. return (borel-of mtopology) x) ‘ M))
                                   (λx. return (borel-of mtopology) x) inv
    by(simp add: s.mtopology-submetric homeomorphic-maps-def)
qed
```

```
corollary homeomorphic-space-mtopology-return:
  mtopology homeomorphic-space (subtopology LPm.mtopology ((λx. return (borel-of
mtopology) x) ‘ M))
  using homeomorphic-map-return homeomorphic-space by fast

lemma closedin-returnM: closedin LPm.mtopology ((λx. return (borel-of mtopol-
ogy) x) ‘ M)
  unfolding LPm.metric-closedin-iff-sequentially-closed
proof safe
  fix Ni N
 assume h:range Ni ⊆ (λx. return (borel-of mtopology) x) ‘ M limitin LPm.mtopology
Ni N sequentially
  from range-subsetD[OF this(1)]
  obtain xi where xi: ⋀i. xi i ∈ M Ni = (λi. return (borel-of mtopology) (xi i))
    unfolding image-def by simp metis
  have sets-N[measurable-cong]: sets N = sets (borel-of mtopology)
    by (meson LPm.limitin-mspace h(2) inP-D)
  have [measurable]:⋀n. {xi n} ∈ sets N
   by (simp add: Hausdorff-space-mtopology borel-of-closed closedin-Hausdorff-sing-eq
sets-N xi(1))
  interpret N: finite-measure N
    by (meson LPm.limitin-metric-dist-null h(2) inP-D(1))
  interpret Ni: prob-space Ni i for i
    by(auto intro!: prob-space-return simp: xi space-borel-of)
  have N-r: ereal (measure N A) ≤ ereal 1 for A
    unfolding ereal-less-eq(3)
  proof(rule order.trans[OF N.bounded-measure])
    interpret mweak-conv-fin M d Ni N sequentially
        using limitin-topspace[OF h(2)] by(auto intro!: inP-mweak-conv-fin inP-I
return-inP simp: xi(2))
    have mweak-conv-seq Ni N
      using converge-imp-mweak-conv h(2) xi(2) by force
    from mweak-conv-imp-limit-space[OF this]
    show measure N (space N) ≤ 1
     by(auto intro!: tendsto-upperbound[where F=sequentially and f=λn. Ni.prob
n (space N)] simp: space-N space-Ni)
  qed
  have ∃x. limitin mtopology xi x sequentially
  proof(rule ccontr)
    assume contr:∄x. limitin mtopology xi x sequentially
    have MCauchy-xi: MCauchy xi
      using MCauchy-iff-MCauchy-return[THEN iffD2,of xi,
          OF - LPm.convergent-imp-MCauchy[OF - h(2)[simplified xi(2)]]] xi
      by fastforce
    have 0:∄x. limitin mtopology (xi ∘ a) x sequentially if a: strict-mono a for a
:: nat ⇒ nat
      using MCauchy-convergent-subsequence[OF MCauchy-xi a] contr by blast
    have inf: infinite (range xi)
```

by (*metis 0 Bolzano-Weierstrass-property MCauchy-xi MCauchy-def finite-subset preorder-class.order.refl*)
have *cl*: *closedin mtopology* (*range* (*xi* ∘ *a*)) **if** *a*: *strict-mono a* **for** *a* :: *nat* ⇒ *nat*
unfolding *closedin-metric*
proof *safe*
fix *x*
assume *x*:*x* ∈ *M x* ∉ *range* (*xi* ∘ *a*)
from *0 a* **have** ¬ *limitin mtopology* (*xi* ∘ *a*) *x sequentially*
by *blast*
then obtain *e* **where** *e*: *e* > *0* ⋀*n0*. ∃ *n*≥*n0*. *d* ((*xi* ∘ *a*) *n*) *x* ≥ *e*
using *xi*(*1*) *x* **by**(*fastforce simp*: *limit-metric-sequentially*)
then obtain *n0* **where** *n0*: ⋀*n m*. *n* ≥ *n0* ⟹ *m* ≥ *n0* ⟹ *d* ((*xi* ∘ *a*) *n*) ((*xi* ∘ *a*) *m*) < *e* / *2*
using *MCauchy-subsequence*[*OF a MCauchy-xi*]
by (*meson MCauchy-def zero-less-divide-iff zero-less-numeral*)
obtain *n1* **where** *n1*: *n1* ≥ *n0 d* ((*xi* ∘ *a*) *n1*) *x* ≥ *e*
using *e*(*2*) **by** *blast*
define *e′* **where** *e′* ≡ *Min* ((λ*n*. *d x* ((*xi* ∘ *a*) *n*))‘ {..*n0*})
have *e′-pos*: *e′* > *0*
unfolding *e′-def* **using** *x xi*(*1*) **by**(*subst linorder-class.Min-gr-iff*) *auto*
have *d x* ((*xi* ∘ *a*) *n*) ≥ *min* (*e* / *2*) *e′* **for** *n*
proof(*cases n* ≤ *n0*)
assume ¬ *n* ≤ *n0*
then have *d* ((*xi* ∘ *a*) *n*) ((*xi* ∘ *a*) *n1*) < *e* / *2*
using *n1*(*1*) *n0* **by** *simp*
hence *e* / *2* ≤ *d x* ((*xi* ∘ *a*) *n1*) − *d* ((*xi* ∘ *a*) *n*) ((*xi* ∘ *a*) *n1*)
using *n1*(*2*) **by**(*simp add*: *commute*)
also have ... ≤ *d x* ((*xi* ∘ *a*) *n*)
using *triangle*[*OF x*(*1*) *xi*(*1*)[*of a n*] *xi*(*1*)[*of a n1*]] **by** *simp*
finally show *?thesis*
by *simp*
qed(*auto intro*!: *linorder-class.Min-le min.coboundedI2 simp*: *e′-def*)
thus ∃ *r*>*0*. *disjnt* (*range* (*xi* ∘ *a*)) (*mball x r*)
using *e′-pos e*(*1*) *x*(*1*) *xi*(*1*) *linorder-not-less*
by(*fastforce intro*!: *exI*[**where** *x*=*min* (*e* / *2*) *e′*] *simp*: *disjnt-def simp del*: *min-less-iff-conj*)
qed(*use xi* **in** *auto*)
hence *meas*: *strict-mono a* ⟹ (*range* (*xi* ∘ *a*)) ∈ *sets* (*borel-of mtopology*) **for** *a* :: *nat* ⇒ *nat*
by(*auto simp*: *borel-of-closed*)
have *1*:*measure N* (*range* (*xi* ∘ *a*)) = *1* **if** *a*: *strict-mono a* **for** *a* :: *nat* ⇒ *nat*
proof −
interpret *mweak-conv-fin M d Ni N sequentially*
using *limitin-topspace*[*OF h*(*2*)] *xi*(*1*) **by**(*auto intro*!: *inP-mweak-conv-fin simp*: *xi*(*2*))
have *mweak-conv-seq Ni N*
using *converge-imp-mweak-conv*[*OF h*(*2*)] *xi*(*2*) **by** *simp*
hence ∗: *closedin mtopology A* ⟹ *limsup* (λ*n*. *ereal* (*measure* (*Ni n*) *A*)) ≤

ereal (*measure N A*) **for** *A*
    using *mweak-conv-eq2* **by** *blast*
  have *ereal 1* ≤ *limsup* (λ*n. ereal* (*measure* (*Ni n*) (*range* (*xi* ∘ *a*))))
    using *meas*[*OF a*] *seq-suble*[*OF a*]
      by(*auto simp*: *limsup-INF-SUP le-Inf-iff le-Sup-iff xi*(*2*) *measure-return indicator-def one-ereal-def*)
  also have ... ≤ *ereal* (*measure N* (*range* (*xi* ∘ *a*)))
    by(*intro* ∗ *a cl*)
  finally show *?thesis*
    using *N-r* **by**(*auto intro*!: *antisym*)
qed
have *2*:*measure N* {*xi n*} = *0* **for** *n*
proof −
  have *infinite* {*i. xi i* ≠ *xi n*}
  proof
    assume *finite* {*i. xi i* ≠ *xi n*}
    then have *finite* (*xi* ‘ {*i. xi i* ≠ *xi n*})
      by *blast*
    moreover have (*xi* ‘ {*i. xi i* ≠ *xi n*}) = *range xi* − {*xi n*}
      by *auto*
    ultimately show *False*
      using *inf* **by** *auto*
  qed
  from *infinite-enumerate*[*OF this*]
  obtain *a* :: *nat* ⇒ *nat* **where** *r*: *strict-mono a* ⋀*i. a i* ∈ {*i. xi i* ≠ *xi n*}
    by *blast*
  hence *disj*: *range* (*xi* ∘ *a*) ∩ {*xi n*} = {}
    by *fastforce*
  from *N.finite-measure-Union*[*OF - - this*]
  have *measure N* (*range* (*xi* ∘ *a*) ∪ {*xi n*}) = *1* + *measure N* {*xi n*}
    using *meas*[*OF r*(*1*)] *1*[*OF r*(*1*)] **by** *simp*
  thus *?thesis*
    using *N-r*[*of range* (*xi* ∘ *a*) ∪ {*xi n*}] *measure-nonneg*[*of N* {*xi n*}] **by** *simp*
qed
have *measure N* (*range xi*) = *0*
proof −
  have *count*: *countable* (*range xi*)
    by *blast*
  define *Xn* **where** *Xn* ≡ (λ*n.* {*from-nat-into* (*range xi*) *n*})
  have *Un-Xn*: *range xi* = (⋃ *n. Xn n*)
  using *bij-betw-from-nat-into*[*OF count inf*] **by** (*simp add*: *UNION-singleton-eq-range Xn-def*)
  have *disjXn*: *disjoint-family Xn*
  using *bij-betw-from-nat-into*[*OF count inf*] **by** (*simp add*: *inf disjoint-family-on-def Xn-def*)
  have [*measurable*]: ⋀*n. Xn n* ∈ *sets N*
    using *bij-betw-from-nat-into*[*OF count inf*]
      by (*metis UNIV-I Xn-def* ‹⋀*n.* {*xi n*} ∈ *sets N*› *bij-betw-iff-bijections image-iff*)

have *eq0*: ⋀*n. measure N* (*Xn n*) = *0*
by (*metis bij-betw-from-nat-into*[*OF count inf*] *2 UNIV-I Xn-def bij-betw-imp-surj-on image-iff*)
have *measure N* (*range xi*) = *measure N* (⋃*n. Xn n*)
by(*simp add*: *Un-Xn*)
also have ... = (∑*n. measure N* (*Xn n*))
using *N.suminf-measure*[*OF - disjXn*] **by** *fastforce*
also have ... = *0*
by(*simp add*: *eq0*)
finally show *?thesis* **.**
qed
with *1*[*OF strict-mono-id*] **show** *False* **by** *simp*
qed
then obtain *x* **where** *x*: *limitin mtopology xi x sequentially*
by *blast*
show *N* ∈ (λ*x. return* (*borel-of mtopology*) *x*) ‘ *M*
using *limitin-topspace*[*OF x*] **by**(*simp add*: *LPm.limitin-metric-unique*[*OF h*(*2*)[*simplified xi*(*2*)] *conv-conv-return*[*OF x*]])
qed *simp*

corollary *separable-iff-LPm-separable*: *separable-space mtopology* ⟷ *separable-space LPm.mtopology*
using *homeomorphic-space-second-countability*[*OF homeomorphic-space-mtopology-return*] *separable-LPm*
by(*auto simp*: *separable-space-iff-second-countable LPm.separable-space-iff-second-countable second-countable-subtopology*)

corollary *LPmcomplete-mcomplete*:
assumes *LPm.mcomplete*
shows *mcomplete*
unfolding *mcomplete-def*
proof *safe*
fix *xn*
assume *h*: *MCauchy xn*
hence *1*: *range xn* ⊆ *M*
using *MCauchy-def* **by** *blast*
interpret *Submetric* 𝒫 *LPm* (λ*x. return* (*borel-of mtopology*) *x*) ‘ *M*
by (*metis LPm.Metric-space-axioms LPm.topspace-mtopology Submetric.intro Submetric-axioms.intro closedin-returnM closedin-subset*)
have *sub.mcomplete*
using *assms*(*1*) *closedin-eq-mcomplete closedin-returnM* **by** *blast*
moreover have *sub.MCauchy* (λ*n. return* (*borel-of mtopology*) (*xn n*))
using *MCauchy-iff-MCauchy-return*[*OF 1*] *1* **by** (*simp add*: *MCauchy-submetric h image-subset-iff*)
ultimately obtain *x* **where**
x: *x* ∈ *M limitin LPm.mtopology* (λ*n. return* (*borel-of mtopology*) (*xn n*))
(*return* (*borel-of mtopology*) *x*) *sequentially*
unfolding *sub.mcomplete-def limitin-submetric-iff* **by** *blast*
thus ∃*x. limitin mtopology xn x sequentially*

by(*auto simp*: *conv-iff-conv-return*[*OF 1 x*(*1*),*symmetric*])
qed

corollary *mcomplete-iff-LPmcomplete*: *separable-space mtopology* ⟹ *mcomplete* ⟷ *LPm.mcomplete*
by(*auto simp add*: *mcomplete-LPmcomplete LPmcomplete-mcomplete*)

lemma *LPmcompact-imp-mcompact*: *compact-space LPm.mtopology* ⟹ *compact-space mtopology*
by (*meson closedin-compact-space closedin-returnM compact-space-subtopology homeomorphic-compact-space homeomorphic-space-mtopology-return*)

end

corollary *Polish-space-weak-conv-topology*:
assumes *Polish-space X*
shows *Polish-space* (*weak-conv-topology X*)
proof −
obtain *d* **where** *d*:*Metric-space* (*topspace X*) *d Metric-space.mcomplete* (*topspace X*) *d*
Metric-space.mtopology (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms completely-metrizable-space-def Polish-space-imp-completely-metrizable-space*)
then interpret *Levy-Prokhorov topspace X d*
by(*auto simp*: *Levy-Prokhorov-def*)
have *separable-space mtopology*
by (*simp add*: *assms d*(*3*) *Polish-space-imp-separable-space*)
thus *?thesis*
using *LPm.Polish-space-mtopology LPmtopology-eq-weak-conv-topology d*(*2*) *d*(*3*) *mcomplete-LPmcomplete separable-LPm* **by** *force*
qed

5.4 Prokhorov Theorem for Topology of Weak Convergence

lemma *relatively-compact-imp-tight*:
assumes *Polish-space X* Γ ⊆ {*N*. *sets N* = *sets* (*borel-of X*) ∧ *finite-measure N*}
and *compactin* (*weak-conv-topology X*) (*weak-conv-topology X closure-of* Γ)
shows *tight-on-set X* Γ
proof −
obtain *d* **where** *d*:*Metric-space* (*topspace X*) *d Metric-space.mcomplete* (*topspace X*) *d*
Metric-space.mtopology (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms*(*1*) *completely-metrizable-space-def Polish-space-imp-completely-metrizable-space*)
note *sep* = *Polish-space-imp-separable-space*[*OF assms*(*1*)]
hence *sep'*:*separable-space* (*Metric-space.mtopology* (*topspace X*) *d*)
by(*simp add*: *d*)
interpret *Levy-Prokhorov topspace X d*

by(*auto simp*: *d Levy-Prokhorov-def*)
show *?thesis*
using *relatively-compact-imp-tight-LP*[*of* Γ] *assms sep inP-iff*
by(*fastforce simp add*: *d LPmtopology-eq-weak-conv-topology*[*OF sep'*])
qed

lemma *tight-imp-relatively-compact*:
assumes *metrizable-space X separable-space X*
$\Gamma \subseteq \{N.\ N\ (space\ N) \leq ennreal\ r \wedge sets\ N = sets\ (borel\text{-}of\ X)\}$
and *tight-on-set X* Γ
shows *compactin* (*weak-conv-topology X*) (*weak-conv-topology X closure-of* Γ)
proof −
obtain *d* **where** *d*:*Metric-space* (*topspace X*) *d Metric-space.mtopology* (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms*(*1*) *metrizable-space-def*)
hence *sep'*:*separable-space* (*Metric-space.mtopology* (*topspace X*) *d*)
by(*simp add*: *d assms*)
show *?thesis*
proof(*cases* $r \leq 0$)
assume $r \leq 0$
then have $\{N.\ N\ (space\ N) \leq ennreal\ r \wedge sets\ N = sets\ (borel\text{-}of\ X)\} = \{null\text{-}measure\ (borel\text{-}of\ X)\}$
by(*fastforce simp*: *ennreal-neg le-zero-eq*[*THEN iffD1*,*OF order.trans*[*OF emeasure-space*]] *intro*!: *measure-eqI*)
then have $\Gamma = \{\} \vee \Gamma = \{null\text{-}measure\ (borel\text{-}of\ X)\}$
using *assms*(*3*) **by** *auto*
moreover have *weak-conv-topology X closure-of* {*null-measure* (*borel-of X*)} = {*null-measure* (*borel-of X*)}
by(*intro closure-of-eq*[*THEN iffD2*] *closedin-Hausdorff-singleton metrizable-imp-Hausdorff-space metrizable-space-subtopology metrizable-weak-conv-topology assms*)
(*auto intro*!: *finite-measureI*)
ultimately show *?thesis*
by (*auto intro*!: *finite-measureI*)
next
assume $\neg\ r \leq 0$
then interpret *Levy-Prokhorov topspace X d*
by(*auto simp*: *d Levy-Prokhorov-def*)
show *?thesis*
using *tight-imp-relatively-compact-LP*[*of* Γ] *assms*
by(*auto simp add*: *d LPmtopology-eq-weak-conv-topology*[*OF sep'*])
qed
qed

lemma *Prokhorov*:
assumes *Polish-space X* $\Gamma \subseteq \{N.\ N\ (space\ N) \leq ennreal\ r \wedge sets\ N = sets\ (borel\text{-}of\ X)\}$
shows *tight-on-set X* Γ ⟷ *compactin* (*weak-conv-topology X*) (*weak-conv-topology X closure-of* Γ)
proof −

```
  have Γ ⊆ {N. sets N = sets (borel-of X) ∧ finite-measure N}
    using assms(2) by(auto intro!: finite-measureI simp: top.extremum-unique)
  thus ?thesis
    using relatively-compact-imp-tight tight-imp-relatively-compact assms
        Polish-space-imp-metrizable-space Polish-space-imp-separable-space
    by (metis (mono-tags, lifting))
qed

corollary tight-on-set-imp-convergent-subsequence:
  fixes Ni :: nat ⇒ - measure
  assumes metrizable-space X separable-space X
    and tight-on-set X (range Ni) ⋀i. (Ni i) (space (Ni i)) ≤ ennreal r
  shows ∃ a N. strict-mono a ∧ finite-measure N ∧ sets N = sets (borel-of X)
            ∧ N (space N) ≤ ennreal r ∧ weak-conv-on (Ni ∘ a) N sequentially X
proof(cases r ≤ 0)
  case True
  then have Ni = (λi. null-measure (borel-of X))
    using assms(3)  order.trans[OF emeasure-space assms(4)]
    by(auto simp: tight-on-set-def ennreal-neg intro!: measure-eqI)
  thus ?thesis
    using weak-conv-on-const[of Ni]
      by(auto intro!: exI[where x=id] exI[where x=null-measure (borel-of X)]
strict-mono-id finite-measureI)
next
  case False
  then have r[arith]:r > 0 by linarith
 obtain d where d: Metric-space (topspace X) d Metric-space.mtopology (topspace
X) d = X
    by (metis Metric-space.topspace-mtopology assms(1) metrizable-space-def)
  then interpret d: Metric-space topspace X d
    by blast
  interpret Levy-Prokhorov topspace X d
    by(auto simp: Levy-Prokhorov-def d )
 have range-Ni: range Ni ⊆ {N. N (space N) ≤ ennreal r ∧ sets N = sets (borel-of
X)}
    using assms(3,4) by(auto simp: tight-on-set-def)
  hence Ni-fin: ⋀i. finite-measure (Ni i)
    by (meson assms(3) range-eqI tight-on-set-def)
  have range-Ni':LPm.mtopology closure-of range Ni
              ⊆ {N. N (space N) ≤ ennreal r ∧ sets N = sets (borel-of X)}
   by (metis (no-types, lifting) Collect-cong closedin-bounded-measures closure-of-minimal
d(2) range-Ni)
  have compactin LPm.mtopology (LPm.mtopology closure-of (range Ni))
     using assms(2,3) range-Ni by(auto intro!: tight-imp-relatively-compact-LP
simp: d(2))
  from LPm.compactin-sequentially[THEN iffD1,OF this] range-Ni
  obtain a N where N ∈ LPm.mtopology closure-of range Ni strict-mono a
    limitin LPm.mtopology (Ni ∘ a) N sequentially
   by (metis (no-types, lifting) LPm.topspace-mtopology assms(3) closure-of-subset
```

d(2) *inP-I subsetI tight-on-set-def*)
moreover hence *finite-measure N sets N* = *sets* (*borel-of X*) *N* (*space N*) ≤ *ennreal r*
using *range-Ni′* **by** (*auto simp add*: *LPm.limitin-metric inP-iff*)
ultimately show *?thesis*
using *range-Ni Ni-fin assms*(4)
by(*fastforce intro*!: *converge-imp-mweak-conv*[*simplified d*] *exI*[**where** *x*=*a*] *exI*[**where** *x*=*N*] *inP-I*
simp: *image-subset-iff d*(2))
qed

end

theory *Space-of-Finite-Measures*
imports *Prokhorov-Theorem*
begin

6 Measurable Space of Finite Measures

6.1 Measurable Space of Finite Measures

We define the measurable space of all finite measures in the same way as *subprob-algebra*.

definition *finite-measure-algebra* :: ′*a measure* ⇒ ′*a measure measure* **where**
finite-measure-algebra K =
(*SUP A* ∈ *sets K*. *vimage-algebra* {*M*. *finite-measure M* ∧ *sets M* = *sets K*} (*λM*. *emeasure M A*) *borel*)

lemma *space-finite-measure-algebra*:
space (*finite-measure-algebra A*) = {*M*. *finite-measure M* ∧ *sets M* = *sets A*}
by (*auto simp add*:*finite-measure-algebra-def space-Sup-eq-UN*)

lemma *finite-measure-algebra-cong*: *sets M* = *sets N* =⇒ *finite-measure-algebra M* = *finite-measure-algebra N*
by (*simp add*: *finite-measure-algebra-def*)

lemma *measurable-emeasure-finite-measure-algebra*[*measurable*]:
a ∈ *sets A* =⇒ (*λM*. *emeasure M a*) ∈ *borel-measurable* (*finite-measure-algebra A*)
by (*auto intro*!: *measurable-Sup1 measurable-vimage-algebra1 simp*: *finite-measure-algebra-def*)

lemma *measurable-measure-finite-measure-algebra*[*measurable*]:
a ∈ *sets A* =⇒ (*λM*. *measure M a*) ∈ *borel-measurable* (*finite-measure-algebra A*)
unfolding *measure-def* **by** *measurable*

lemma *finite-measure-measurableD*:
assumes *N*: *N* ∈ *measurable M* (*finite-measure-algebra S*) **and** *x*: *x* ∈ *space M*

shows *space* (*N x*) = *space S*
and *sets* (*N x*) = *sets S*
and *measurable* (*N x*) *K* = *measurable S K*
and *measurable K* (*N x*) = *measurable K S*
using *measurable-space*[*OF N x*]
by (*auto simp*: *space-finite-measure-algebra intro*!: *measurable-cong-sets dest*: *sets-eq-imp-space-eq*)

ML ‹

```
fun finite-measure-cong thm ctxt = (
  let
    val thm' = Thm.transfer' ctxt thm
    val free = thm' |> Thm.concl-of |> HOLogic.dest-Trueprop |> dest-comb |> fst
|>
      dest-comb |> snd |> strip-abs-body |> head-of |> is-Free
  in
    if free then ([], Measurable.add-local-cong (thm' RS @{thm finite-measure-measurableD(2)})
ctxt)
          else ([], ctxt)
  end
  handle THM - => ([], ctxt) | TERM - => ([], ctxt))
```

›

setup ‹
Context.theory-map (*Measurable.add-preprocessor finite-measure-cong subprob-cong*)
›

context
fixes *K M N* **assumes** *K*: *K* ∈ *measurable M* (*finite-measure-algebra N*)
begin

lemma *finite-measure-space-kernel*: *a* ∈ *space M* ⟹ *finite-measure* (*K a*)
using *measurable-space*[*OF K*] **by** (*simp add*: *space-finite-measure-algebra*)

lemma *sets-finite-kernel*: *a* ∈ *space M* ⟹ *sets* (*K a*) = *sets N*
using *measurable-space*[*OF K*] **by** (*simp add*: *space-finite-measure-algebra*)

lemma *measurable-emeasure-finite-kernel*[*measurable*]:
A ∈ *sets N* ⟹ (λ*a*. *emeasure* (*K a*) *A*) ∈ *borel-measurable M*
using *measurable-compose*[*OF K measurable-emeasure-finite-measure-algebra*] **.**

end

lemma *measurable-finite-measure-algebra*:
(⋀*a*. *a* ∈ *space M* ⟹ *finite-measure* (*K a*)) ⟹
(⋀*a*. *a* ∈ *space M* ⟹ *sets* (*K a*) = *sets N*) ⟹
(⋀*A*. *A* ∈ *sets N* ⟹ (λ*a*. *emeasure* (*K a*) *A*) ∈ *borel-measurable M*) ⟹

K ∈ *measurable M* (*finite-measure-algebra N*)
by (*auto intro*!: *measurable-Sup2 measurable-vimage-algebra2 simp*: *finite-measure-algebra-def*)

lemma *measurable-finite-markov*:
K ∈ *measurable M* (*finite-measure-algebra M*) ⟷
(∀ *x*∈*space M*. *finite-measure* (*K x*) ∧ *sets* (*K x*) = *sets M*) ∧
(∀ *A*∈*sets M*. (λ*x*. *emeasure* (*K x*) *A*) ∈ *measurable M borel*)
proof
assume (∀ *x*∈*space M*. *finite-measure* (*K x*) ∧ *sets* (*K x*) = *sets M*) ∧
(∀ *A*∈*sets M*. (λ*x*. *emeasure* (*K x*) *A*) ∈ *borel-measurable M*)
then show *K* ∈ *measurable M* (*finite-measure-algebra M*)
by (*intro measurable-finite-measure-algebra*) *auto*
next
assume *K* ∈ *measurable M* (*finite-measure-algebra M*)
then show (∀ *x*∈*space M*. *finite-measure* (*K x*) ∧ *sets* (*K x*) = *sets M*) ∧
(∀ *A*∈*sets M*. (λ*x*. *emeasure* (*K x*) *A*) ∈ *borel-measurable M*)
by (*auto dest*: *finite-measure-space-kernel sets-finite-kernel*)
qed

lemma *measurable-finite-measure-algebra-generated*:
assumes *eq*: *sets N* = *sigma-sets* Ω *G* **and** *Int-stable G G* ⊆ *Pow* Ω
assumes *subsp*: ⋀*a*. *a* ∈ *space M* ⟹ *finite-measure* (*K a*)
assumes *sets*: ⋀*a*. *a* ∈ *space M* ⟹ *sets* (*K a*) = *sets N*
assumes ⋀*A*. *A* ∈ *G* ⟹ (λ*a*. *emeasure* (*K a*) *A*) ∈ *borel-measurable M*
assumes Ω: (λ*a*. *emeasure* (*K a*) Ω) ∈ *borel-measurable M*
shows *K* ∈ *measurable M* (*finite-measure-algebra N*)
proof (*rule measurable-finite-measure-algebra*)
fix *a* **assume** *a* ∈ *space M* **then show** *finite-measure* (*K a*) *sets* (*K a*) = *sets N* **by** *fact*+
next
interpret *G*: *sigma-algebra* Ω *sigma-sets* Ω *G*
using ‹*G* ⊆ *Pow* Ω› **by** (*rule sigma-algebra-sigma-sets*)
fix *A* **assume** *A* ∈ *sets N* **with** *assms*(*2*,*3*) **show** (λ*a*. *emeasure* (*K a*) *A*) ∈ *borel-measurable M*
unfolding ‹*sets N* = *sigma-sets* Ω *G*›
proof (*induction rule*: *sigma-sets-induct-disjoint*)
case (*basic A*) **then show** *?case* **by** *fact*
next
case *empty* **then show** *?case* **by** *simp*
next
case (*compl A*)
have (λ*a*. *emeasure* (*K a*) (Ω − *A*)) ∈ *borel-measurable M* ⟷
(λ*a*. *emeasure* (*K a*) Ω − *emeasure* (*K a*) *A*) ∈ *borel-measurable M*
using *G.top G.sets-into-space sets eq compl finite-measure.emeasure-finite*[*OF subsp*]
by (*intro measurable-cong emeasure-Diff*) *auto*
with *compl* Ω **show** *?case*
by *simp*
next

case (*union F*)
moreover have ($\lambda a.$ *emeasure* (K a) ($\bigcup i.$ F i)) $\in$ *borel-measurable M* $\longleftrightarrow$
($\lambda a.$ $\sum i.$ *emeasure* (K a) (F i)) $\in$ *borel-measurable M*
using *sets union eq*
by (*intro measurable-cong suminf-emeasure*[*symmetric*]) *auto*
ultimately show *?case*
by *auto*
qed
qed

lemma *space-finite-measure-algebra-empty*: *space* $N = \{\} \Longrightarrow$ *space* (*finite-measure-algebra N*) = {*null-measure N*}
by(*fastforce simp*: *space-finite-measure-algebra space-empty-iff intro*!: *measure-eqI finite-measureI*)

lemma *sets-subprob-algebra-restrict*:
sets (*subprob-algebra M*) = *sets* (*restrict-space* (*finite-measure-algebra M*) {*N. subprob-space N*})
(**is** *sets ?L* = *sets ?R*)
proof −
have *1*:*id* $\in$ *measurable ?L ?R*
using *sets.sets-into-space*[*of - M*]
by(*auto intro*!: *measurable-restrict-space2 Int-stableI*
measurable-finite-measure-algebra-generated[**where** Ω=*space M* **and** *G*=*sets M*]
simp: *space-subprob-algebra subprob-space-def sets.sigma-sets-eq*)
have *2*:*id* $\in$ *measurable ?R ?L*
using *sets.sets-into-space*[*of - M*]
by(*auto intro*!: *measurable-subprob-algebra-generated*[**where** Ω=*space M* **and** *G*=*sets M*] *Int-stableI*
simp: *sets.sigma-sets-eq space-restrict-space space-finite-measure-algebra measurable-restrict-space1*)
have *3*: *space ?L* = *space ?R*
by(*auto simp*: *space-restrict-space space-subprob-algebra space-finite-measure-algebra subprob-space-def*)
have [*simp*]:$\bigwedge A.$ $A \in$ *sets ?L* $\Longrightarrow A \cap$ *space ?R* $= A$ $\bigwedge A.$ $A \in$ *sets ?R* $\Longrightarrow A \cap$ *space ?L* $= A$
using *3 sets.sets-into-space* **by** *auto*
show *?thesis*
using *measurable-sets*[*OF 1*] *measurable-sets*[*OF 2*] **by** *auto*
qed

6.2 Equivalence between Spaces of Finite Measures

Corollary 17.21 [2].

lemma(**in** *Levy-Prokhorov*) *openin-lower-semicontinuous*:
assumes *openin mtopology U*
shows *lower-semicontinuous-map LPm.mtopology* ($\lambda N.$ *measure N U*)
unfolding *lower-semicontinuous-map-liminf-real*[*OF LPm.first-countable-mtopology*]

proof *safe*
  fix *Ni N*
  assume *h*:*limitin LPm.mtopology Ni N sequentially*
  then obtain *K* **where** *K*: $\bigwedge n.\ n \geq K \Longrightarrow Ni\ n \in \mathcal{P}$
    by(*simp add*: *mtopology-of-def LPm.limit-metric-sequentially*)
      (*meson LPm.mbounded-alt-pos LPm.mbounded-empty*)
  have *h'*: *limitin LPm.mtopology* ($\lambda n.$ *Ni* ($n + K$)) *N sequentially*
    by (*simp add*: *h limitin-sequentially-offset*)
  interpret *mweak-conv-fin M d* $\lambda n.$ *Ni* ($n + K$) *N sequentially*
      using *K h* **by**(*auto intro*!: *inP-mweak-conv-fin simp*: *mtopology-of-def dest*: *LPm.limitin-mspace*)
  have *mweak-conv-seq* ($\lambda n.$ *Ni* ($n + K$)) *N*
    using *K LPm.Self-def converge-imp-mweak-conv h'* **by** *auto*
  hence *ereal* (*measure N U*) $\leq$ *liminf* ($\lambda x.$ *ereal* (*measure* (*Ni* ($x + K$)) *U*))
    using *assms* **by**(*simp add*: *mweak-conv-eq3*)
  thus *ereal* (*measure N U*) $\leq$ *liminf* ($\lambda x.$ *ereal* (*measure* (*Ni x*) *U*))
    unfolding *liminf-shift-k*[*of* $\lambda x.$ *ereal* (*measure* (*Ni x*) *U*) *K*] **.**
qed

lemma(**in** *Levy-Prokhorov*) *closedin-upper-semicontinuous*:
  assumes *closedin mtopology A*
  shows *upper-semicontinuous-map LPm.mtopology* ($\lambda N.$ *measure N A*)
  unfolding *upper-semicontinuous-map-limsup-real*[*OF LPm.first-countable-mtopology*]
proof *safe*
  fix *Ni N*
  assume *h*:*limitin LPm.mtopology Ni N sequentially*
  then obtain *K* **where** *K*: $\bigwedge n.\ n \geq K \Longrightarrow Ni\ n \in \mathcal{P}$
    by(*simp add*: *mtopology-of-def LPm.limit-metric-sequentially*)
      (*meson LPm.mbounded-alt-pos LPm.mbounded-empty*)
  have *h'*: *limitin LPm.mtopology* ($\lambda n.$ *Ni* ($n + K$)) *N sequentially*
    by (*simp add*: *h limitin-sequentially-offset*)
  interpret *mweak-conv-fin M d* $\lambda n.$ *Ni* ($n + K$) *N sequentially*
      using *K h* **by**(*auto intro*!: *inP-mweak-conv-fin simp*: *mtopology-of-def dest*: *LPm.limitin-mspace*)
  have *mweak-conv-seq* ($\lambda n.$ *Ni* ($n + K$)) *N*
    using *K LPm.Self-def converge-imp-mweak-conv h'* **by** *auto*
  hence *limsup* ($\lambda x.$ *ereal* (*measure* (*Ni* ($x + K$)) *A*)) $\leq$ *ereal* (*measure N A*)
    using *assms* **by**(*auto simp*: *mweak-conv-eq2*)
  thus *limsup* ($\lambda x.$ *ereal* (*measure* (*Ni x*) *A*)) $\leq$ *ereal* (*measure N A*)
    unfolding *limsup-shift-k*[*of* $\lambda x.$ *ereal* (*measure* (*Ni x*) *A*) *K*] **.**
qed

context *Levy-Prokhorov*
begin

We show that the measurable space generated from *LPm.mtopology* is equal to *finite-measure-algebra* (*borel-of LPm.mtopology*).

lemma *sets-LPm1*: *sets* (*finite-measure-algebra* (*borel-of mtopology*))
          $\subseteq$ *sets* (*borel-of LPm.mtopology*) (**is** *sets ?Giry* $\subseteq$ *sets ?Levy*)

proof *safe*
have *space-eq*: *space ?Levy* = *space ?Giry*
by(*simp add*: *space-finite-measure-algebra space-borel-of*) (*auto simp add*: $\mathcal{P}$-*def*)
have *1*:⋀*A*. *openin mtopology A* ⟹ (λ*N*. *measure N A*) ∈ *borel-measurable ?Levy*
by(*auto intro*!: *lower-semicontinuous-map-measurable openin-lower-semicontinuous*)
have *m*:*id* ∈ *?Levy* $\rightarrow_M$ *?Giry*
proof(*rule measurable-finite-measure-algebra-generated*[**where** Ω=*M* **and** *G*={*U*. *openin mtopology U*}])
show *sets* (*borel-of mtopology*) = *sigma-sets M* {*U*. *openin mtopology U*}
using *sets-borel-of*[*of mtopology*] **by** *simp*
next
show *Int-stable* {*U*. *openin mtopology U*}
by(*auto intro*!: *Int-stableI*)
next
show {*U*. *openin mtopology U*} ⊆ *Pow M*
using *openin-subset*[*of mtopology*] **by** *auto*
next
show ⋀*a*. *a* ∈ *space* (*borel-of LPm.mtopology*) ⟹ *finite-measure* (*id a*)
by(*simp add*: *space-borel-of*) (*simp add*: $\mathcal{P}$-*def*)
next
show ⋀*a*. *a* ∈ *space* (*borel-of LPm.mtopology*) ⟹ *sets* (*id a*) = *sets* (*borel-of mtopology*)
by(*simp add*: *space-borel-of*) (*simp add*: $\mathcal{P}$-*def*)
next
fix *A*
assume *A* ∈ {*U*. *openin mtopology U*}
then have (λ*N*. *measure* (*id N*) *A*) ∈ *borel-measurable* (*borel-of LPm.mtopology*)
by(*simp add*: *1*)
then have *1*:(λ*N*. *ennreal* (*measure* (*id N*) *A*)) ∈ *borel-measurable* (*borel-of LPm.mtopology*)
by *simp*
have *2*:⋀*N*. *N* ∈ *space* (*borel-of LPm.mtopology*) ⟹ *ennreal* (*measure* (*id N*) *A*) = *emeasure* (*id N*) *A*
unfolding *measure-def*
by(*rule ennreal-enn2real*)
(*simp add*: *finite-measure.emeasure-eq-measure space-eq space-finite-measure-algebra*)
show (λ*N*. *emeasure* (*id N*) *A*) ∈ *borel-measurable* (*borel-of LPm.mtopology*)
using *1 measurable-cong*[*THEN iffD1*,*OF 2 1*] **by** *auto*
next
have *openin mtopology M*
by *simp*
then have (λ*N*. *measure* (*id N*) *M*) ∈ *borel-measurable* (*borel-of LPm.mtopology*)
by(*simp add*: *1*)
then have *1*:(λ*N*. *ennreal* (*measure* (*id N*) *M*)) ∈ *borel-measurable* (*borel-of LPm.mtopology*)
by *simp*
have *2*:⋀*N*. *N* ∈ *space* (*borel-of LPm.mtopology*) ⟹ *ennreal* (*measure* (*id N*) *M*) = *emeasure* (*id N*) *M*

```
      unfolding measure-def by(rule ennreal-enn2real)
      (simp add: finite-measure.emeasure-eq-measure space-eq space-finite-measure-algebra)
    show (λN. emeasure (id N) M) ∈ borel-measurable (borel-of LPm.mtopology)
      using 1 measurable-cong[THEN iffD1,OF 2 1] by auto
  qed

  fix A
  assume A:A ∈ sets ?Giry
  from measurable-sets[OF m this] have A ∩ space ?Levy ∈ sets ?Levy
    by simp
  moreover have A ∩ space ?Levy = A
    by (simp add: A space-eq)
  ultimately show A ∈ sets ?Levy
    by simp
qed

lemma sets-LPm2:
  assumes mcomplete separable-space mtopology
   shows sets (borel-of LPm.mtopology) ⊆ sets (finite-measure-algebra (borel-of
mtopology))
    (is sets ?Levy ⊆ sets ?Giry)
proof −
  obtain 𝒪 where base: countable 𝒪 base-in mtopology 𝒪
     using assms(2) second-countable-base-in separable-space-imp-second-countable
by blast
  define funion-of-base where funion-of-base ≡ ⋃ ‘ {U. finite U ∧ U ⊆ 𝒪}
  have funion-of-base-ne: funion-of-base ≠ {}
    by(auto simp: funion-of-base-def)
  have open-funion-of-base: ⋀A. A ∈ funion-of-base ⟹ openin mtopology A
    using base-in-openin[OF base(2)] by(auto simp: funion-of-base-def )
   hence meas-funion-of-base[measurable]: ⋀A. A ∈ funion-of-base ⟹ A ∈ sets
(borel-of mtopology)
    by(auto simp: borel-of-open)
  have countable-funion-of-base: countable funion-of-base
   using countable-Collect-finite-subset[OF base(1)] by(auto simp: funion-of-base-def)

  have sets ?Levy = sigma-sets 𝒫 {LPm.mball a ε |a ε. a ∈ 𝒫 ∧ 0 < ε}
    by(auto simp: borel-of-second-countable'[OF separable-LPm[OF assms(2),
              simplified LPm.separable-space-iff-second-countable]
         base-is-subbase[OF LPm.mtopology-base-in-balls]] intro!: sets-measure-of)
  also have ... = sigma-sets 𝒫 {LPm.mcball a ε |a ε. a ∈ 𝒫 ∧ 0 ≤ ε}
  proof(safe intro!: sigma-sets-eqI)
    fix L and e :: real
    assume h:L ∈ 𝒫 and 0 < e
    have LPm.mball L e = (⋃n. LPm.mcball L (e − 1 / (Suc n)))
    proof safe
      fix N
      assume N: N ∈ LPm.mball L e
      then obtain n where 1 / Suc n < e − LPm L N
```

```
      by (meson LPm.in-mball diff-gt-0-iff-gt nat-approx-posE)
    thus N ∈ (⋃n. LPm.mcball L (e − 1 / real (Suc n)))
      using N by(auto intro!: exI[where x=n] simp: LPm.mcball-def)
  next
    fix N n
    assume N: N ∈ LPm.mcball L (e − 1 / (Suc n))
    with order.strict-trans1[of LPm L N e − 1 / (Suc n) e]
    show N ∈ LPm.mball L e
      by auto
  qed
  also have ... ∈ sigma-sets 𝒫 {LPm.mcball a ε |a ε. a ∈ 𝒫 ∧ 0 ≤ ε}
  proof(rule Union)
    fix n
    consider e − 1 / real (Suc n) < 0 | 0 ≤ e − 1 / real (Suc n) by fastforce
   then show LPm.mcball L (e − 1 / real (Suc n)) ∈ sigma-sets 𝒫 {LPm.mcball
a ε |a ε. a ∈ 𝒫 ∧ 0 ≤ ε}
    proof cases
      case 2
      then show ?thesis
        using h by fast
   qed(use LPm.mcball-eq-empty[of - e − 1 / real (Suc n)] sigma-sets.Empty in
auto)
  qed
  finally show LPm.mball L e ∈ sigma-sets 𝒫 {LPm.mcball a ε |a ε. a ∈ 𝒫 ∧
0 ≤ ε} .
 next
  fix L and e :: real
  assume h:L ∈ 𝒫 0 ≤ e
  have LPm.mcball L e = (⋂n. LPm.mball L (e + 1 / Suc n))
  proof safe
    fix N n
    assume N ∈ LPm.mcball L e
    with order.strict-trans1[of LPm L N e e + 1 / (Suc n)]
    show N ∈ LPm.mball L (e + 1 / (Suc n))
      by auto
  next
    fix N
    assume hn:N ∈ (⋂n. LPm.mball L (e + 1 / real (Suc n)))
    then have N:N ∈ 𝒫
      by auto
    show N ∈ LPm.mcball L e
    proof −
      have LPm L N ≤ e
      proof(rule field-le-epsilon)
        fix l :: real
        assume l > 0
        then obtain n where 1 / (1 + real n) < l
          using nat-approx-posE by auto
        with hn show LPm L N ≤ e + l
```

by(*auto intro*!: *order.trans*[*of LPm L N e* + *1* / (*1* + *real n*) *e* + *l*,*OF less-imp-le*])
qed
thus *?thesis*
using *hn* **by** *auto*
qed
qed
also have ... ∈ *sigma-sets* 𝒫 {*LPm.mball a ε* |*a ε*. *a* ∈ 𝒫 ∧ *0* < *ε*}
proof(*rule sigma-sets-Inter*)
fix *n*
show *LPm.mball L* (*e* + *1* / *real* (*Suc n*)) ∈ *sigma-sets* 𝒫 {*LPm.mball a ε* |*a ε*. *a* ∈ 𝒫 ∧ *0* < *ε*}
using *h* **by**(*auto intro*!: *exI*[**where** *x*=*L*] *exI*[**where** *x*=*e* + *1* / (*1* + *real n*)] *add-nonneg-pos*)
qed *auto*
finally show *LPm.mcball L e* ∈ *sigma-sets* 𝒫 {*LPm.mball a ε* |*a ε*. *a* ∈ 𝒫 ∧ *0* < *ε*} **.**
qed
also have ... = *sigma-sets* (*space ?Giry*) {*LPm.mcball a ε* |*a ε*. *a* ∈ 𝒫 ∧ *0* ≤ *ε*}
unfolding *space-finite-measure-algebra* 𝒫*-def* **by** *meson*
also have ... ⊆ *sets ?Giry*
proof(*rule sigma-sets-le-sets-iff*[*THEN iffD2*])
show {*LPm.mcball a ε* |*a ε*. *a* ∈ 𝒫 ∧ *0* ≤ *ε*} ⊆ *sets ?Giry*
proof *safe*
fix *L* **and** *e* :: *real*
assume *L*:*L* ∈ 𝒫 **and** *e*:*0* ≤ *e*
then have *sets-L*: *sets* (*borel-of mtopology*) = *sets L* **and** *finite-measure L*
by(*auto simp*: *inP-D*)
interpret *L*: *finite-measure L* **by** *fact*
have *LPm.mcball L e*
= (⋂ *A*∈*funion-of-base*.
(⋂ *n*. (λ*N*. *measure N A*) −‘
{..*measure L* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*))} ∩ 𝒫)
∩ (⋂ *n*. (λ*N*. *measure N*
(⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*)))) −‘ {*measure L A* − (*e* + *1* / (*1* + *real n*))..} ∩ 𝒫))
(**is** - = *?rhs*)
unfolding *set-eq-iff*
proof(*intro allI iffI*)
fix *N*
assume *N*: *N* ∈ *LPm.mcball L e*
have *sets-N*: *sets* (*borel-of mtopology*) = *sets N* **and** *finite-measure N*
using *N* **by** *simp-all* (*auto simp*: *inP-D*)
then interpret *N*: *finite-measure N* **by** *simp*
show *N* ∈*?rhs*
proof *safe*
fix *A n*
assume [*measurable*]:*A* ∈ *funion-of-base*

have *LPm L N* < *e* + *1* / (*1* + *real n*)
by(*rule order.strict-trans1*[*of LPm L N e e* + *1* / (*1* + *real n*)]) (*use N* **in** *auto*)
thus *N* ∈ (λ*N*. *measure N A*) −' {..*measure L* (⋃*a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*))}
N ∈ (λ*N*. *measure N* (⋃*a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*)))) −' {*measure L A* − (*e* + *1* / (*1* + *real n*))..}
using *LPm-less-then*[*of L N e* + *1* / (*1* + *real n*) *A*] *N L* **by** *auto*
qed(*use N* **in** *auto*)
next
fix *N*
assume *N* ∈ *?rhs*
then have *N*: *N* ∈ 𝒫
⋀*A n*. *A* ∈ *funion-of-base*
⟹ *measure N A* ≤ *measure L* (⋃*a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*))
⋀*A n*. *A* ∈ *funion-of-base*
⟹ *measure L A* ≤ *measure N* (⋃*a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*))
using *funion-of-base-ne* **by** (*auto simp*: *diff-le-eq*)
then have *sets-N*: *sets* (*borel-of mtopology*) = *sets N*
by(*auto simp*: *inP-D*)
interpret *N*: *finite-measure N*
using *N* **by**(*auto simp*: *inP-D*)
have [*measurable*]: ⋀*A e*. (⋃*a*∈*A*. *mball a e*) ∈ *sets N* ⋀*A e*. (⋃*a*∈*A*. *mball a e*) ∈ *sets L*
by(*auto simp*: *sets-L*[*symmetric*] *sets-N*[*symmetric*])
have *ne*: {*e*. *e* > *0* ∧ (∀*A*∈{*U*. *openin mtopology U*}.
measure L A ≤ *measure N* (⋃*a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃*a*∈*A*. *mball a e*) + *e*)} ≠ {}
using *LPm-ne'*[*OF L.finite-measure-axioms N.finite-measure-axioms*] **by** *fastforce*
have (⨅ {*e*. *e* > *0* ∧ (∀*A*∈{*U*. *openin mtopology U*}.
measure L A ≤ *measure N* (⋃*a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃*a*∈*A*. *mball a e*) + *e*)}) ≤ *e*
proof(*safe intro*!: *cInf-le-iff-less*[**where** *f*=*id*,*simplified*,*THEN iffD2*,*OF ne*])
fix *y*
assume *y*:*e* < *y*
then obtain *n* **where** *1* / *Suc n* < *y* − *e*
by (*meson diff-gt-0-iff-gt nat-approx-posE*)
hence *n*: *e* + *1* / (*1* + *real n*) < *y* **by** *simp*
show ∃ *i*∈{*e*. *0* < *e* ∧ (∀*A*∈{*U*. *openin mtopology U*}.
measure L A ≤ *measure N* (⋃*a*∈*A*. *mball a e*) + *e* ∧
measure N A ≤ *measure L* (⋃*a*∈*A*. *mball a e*) + *e*)}. *i* ≤ *y*
proof(*safe intro*!: *bexI*[**where** *x*=*e* + *1* / (*1* + *real n*)])

fix *A*
assume *A*: *openin mtopology A*
then have *A'*[*measurable*]: *A* ∈ *sets L* *A* ∈ *sets N*
by(*auto simp*: *borel-of-open sets-N*[*symmetric*] *sets-L*[*symmetric*])
have *measure L A* = ⨆ (*measure L* ' {*K*. *compactin mtopology K* ∧ *K* ⊆ *A*})
by(*auto intro*!: *L.inner-regular-Polish*[*OF Polish-space-mtopology*[*OF assms*] *sets-L*])
also have ... ≤ ⨆ (*measure L* ' {*U*. *U* ∈*funion-of-base* ∧ *U* ⊆ *A*})
proof(*safe intro*!: *cSup-mono bdd-aboveI*[**where** *M*=*measure L* (*space L*)] *L.bounded-measure*)
fix *K*
assume *K*:*compactin mtopology K K* ⊆ *A*
obtain 𝒰 **where** *Aun*: *A* = ⋃ 𝒰 𝒰 ⊆ 𝒪
using *A base* **by**(*auto simp*: *base-in-def*)
obtain ℱ **where** *F*: *finite* ℱ ℱ ⊆ 𝒰 *K* ⊆ ⋃ ℱ
using *compactinD*[*OF K*(*1*),*of* 𝒰] *Aun K base-in-openin*[*OF base*(*2*)] **by** *blast*
hence *Ffunion*: ⋃ ℱ ∈ *funion-of-base* ⋃ ℱ ⊆ *A*
using *F Aun K* **by** (*auto simp*: *funion-of-base-def*)
with *F*(*3*) **show** ∃ *a*∈*measure L* ' {*U* ∈ *funion-of-base*. *U* ⊆ *A*}. *measure L K* ≤ *a*
by(*auto intro*!: *exI*[**where** *x*=⋃ ℱ] *L.finite-measure-mono meas-funion-of-base*[*simplified sets-L*])
qed *auto*
also have ... ≤ ⨆ {*measure N* (⋃ *a*∈*U*. *mball a* (*e* + *1* / (*1* + *real n*)))+(*e*+*1* / (*1*+*real n*))
| *U*. *U* ∈ *funion-of-base* ∧ *U* ⊆ *A*}
by(*force intro*!: *cSup-mono N bdd-aboveI*[**where** *M*=*measure N* (*space N*)+(*e* + *1*/(*1*+*real n*))]
N.bounded-measure simp: *funion-of-base-def*)
also have ... ≤ *measure N* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*))
by(*fastforce intro*!:
cSup-le-iff[*THEN iffD2*] *bdd-aboveI*[**where** *M*=*measure N* (*space N*) + (*e* + *1* / (*1* + *real n*))]
N.bounded-measure N.finite-measure-mono
simp: *funion-of-base-def*)
finally show *measure L A*
≤ *measure N* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* / (*1* + *real n*)) **.**
have *measure N A* = ⨆ (*measure N* ' {*K*. *compactin mtopology K* ∧ *K* ⊆ *A*})
by(*auto intro*!: *N.inner-regular-Polish*[*OF Polish-space-mtopology sets-N*] *assms*)
also have ... ≤ ⨆ (*measure N* ' {*U*. *U* ∈ *funion-of-base* ∧ *U* ⊆ *A*})
proof(*safe intro*!: *cSup-mono bdd-aboveI*[**where** *M*=*measure N* (*space N*)] *N.bounded-measure*)
fix *K*

assume *K*:*compactin mtopology K K* ⊆ *A*
obtain $\mathcal{U}$ **where** *Aun*: *A* = ⋃ $\mathcal{U}$ $\mathcal{U}$ ⊆ $\mathcal{O}$
using *A base* **by**(*auto simp*: *base-in-def*)
obtain $\mathcal{F}$ **where** *F*: *finite* $\mathcal{F}$ $\mathcal{F}$ ⊆ $\mathcal{U}$ *K* ⊆ ⋃ $\mathcal{F}$
using *compactinD*[*OF K*(*1*),*of* $\mathcal{U}$] *Aun K base-in-openin*[*OF base*(*2*)]
by *blast*
hence *Ffunion*: ⋃ $\mathcal{F}$ ∈ *funion-of-base* ⋃ $\mathcal{F}$ ⊆ *A*
using *F Aun K* **by** (*auto simp*: *funion-of-base-def*)
with *F*(*3*) **show** ∃ *y*∈ *measure N* ' {*U* ∈ *funion-of-base*. *U* ⊆ *A*}.
measure N K ≤ *y*
by(*auto intro*!: *exI*[**where** *x*=⋃ $\mathcal{F}$] *N*.*finite-measure-mono meas-funion-of-base*[*simplified sets-N*])
qed *auto*
also have ... ≤ ⨆ {*measure L* (⋃ *a*∈*U*. *mball a* (*e* + *1* / (*1* + *real n*)))
+ (*e* + *1* / (*1* + *real n*))
| *U*. *U* ∈ *funion-of-base* ∧ *U* ⊆ *A*}
by(*force intro*!: *cSup-mono N bdd-aboveI*[**where** *M*=*measure L* (*space*
L) + (*e* + *1* / (*1* + *real n*))]
L.*bounded-measure simp*: *funion-of-base-def*)
also have ... ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) +
(*e* + *1* / (*1* + *real n*))
by(*fastforce intro*!:
cSup-le-iff [*THEN iffD2*] *bdd-aboveI*[**where** *M*=*measure L* (*space L*)
+ (*e* + *1* / (*1* + *real n*))]
L.*bounded-measure L*.*finite-measure-mono*
simp: *funion-of-base-def*)
finally show *measure N A* ≤ *measure L* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1*
+ *real n*))) + (*e* + *1* / (*1* + *real n*)) **.**
qed(*insert e n*, *auto intro*!: *add-nonneg-pos*)
qed(*fastforce intro*!: *bdd-belowI*[**where** *m*=*0*])
thus *N* ∈ *LPm*.*mcball L e*
using *N*(*1*) *L* **by**(*auto simp*: *LPm-open*)
qed
also have ... ∈ *sets ?Giry*
proof −
have *h*:(λ*N*. *measure N A*) −'
{..*measure L* (⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) + (*e* + *1* /
(*1* + *real n*))} ∩ $\mathcal{P}$
∈ *sets ?Giry* (**is** *?m1*)
(λ*N*. *measure N*
(⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*)))) −' {*measure L A* − (*e* +
1 / (*1* + *real n*))..} ∩ $\mathcal{P}$
∈ *sets ?Giry* (**is** *?m2*) **if** *A*∈*funion-of-base* **for** *A n*
proof −
have *P*:$\mathcal{P}$ = *space ?Giry* **unfolding** $\mathcal{P}$*-def space-finite-measure-algebra* **by**
auto
have [*measurable*]:*A* ∈ *sets* (*borel-of mtopology*)
(⋃ *a*∈*A*. *mball a* (*e* + *1* / (*1* + *real n*))) ∈ *sets* (*borel-of mtopology*)
using *that* **by** *simp* (*auto intro*!: *borel-of-open*)

show *?m1 ?m2*
by(*auto intro*!: *measurable-sets simp*: *P*)
qed
show *?thesis*
by(*rule sets.countable-INT'*[*OF countable-funion-of-base funion-of-base-ne*])
(*use h* **in** *blast*)
qed
finally show *LPm.mcball L e* ∈ *sets ?Giry* **.**
qed
qed
finally show *?thesis* **.**
qed

corollary *sets-LPm-eq-sets-finite-measure-algebra*:
assumes *mcomplete separable-space mtopology*
shows *sets* (*borel-of LPm.mtopology*) = *sets* (*finite-measure-algebra* (*borel-of mtopology*))
using *sets-LPm1 sets-LPm2*[*OF assms*] **by** *simp*

end

corollary *weak-conv-topology-eq-finite-measure-algebra*:
assumes *Polish-space X*
shows *sets* (*borel-of* (*weak-conv-topology X*)) = *sets* (*finite-measure-algebra* (*borel-of X*))
proof −
obtain *d* **where** *d*:*Metric-space* (*topspace X*) *d Metric-space.mcomplete* (*topspace X*) *d*
Metric-space.mtopology (*topspace X*) *d* = *X*
by (*metis Metric-space.topspace-mtopology assms completely-metrizable-space-def Polish-space-imp-completely-metrizable-space*)
then interpret *Levy-Prokhorov topspace X d*
by (*auto simp add*: *Levy-Prokhorov-def*)
have *sep*: *separable-space mtopology*
by (*simp add*: *assms d*(*3*) *Polish-space-imp-separable-space*)
show *?thesis*
using *sets-LPm-eq-sets-finite-measure-algebra*[*OF d*(*2*) *sep*] *LPmtopology-eq-weak-conv-topology*[*OF sep*]
by(*simp add*: *d*)
qed

corollary *weak-conv-topology-eq-subprob-algebra*:
assumes *Polish-space X*
shows *sets* (*borel-of* (*subtopology* (*weak-conv-topology X*) {*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of X*)}))
= *sets* (*subprob-algebra* (*borel-of X*)) (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = *sets* (*borel-of* (*subtopology* (*weak-conv-topology X*) {*N*. *sets N* = *sets* (*borel-of X*) ∧ *subprob-space N*}))

by *meson*
also have ... = *sets* (*borel-of* (*subtopology* (*weak-conv-topology* *X*) {*N*. *sub-prob-space* *N*}))
using *subtopology-restrict*[*of weak-conv-topology X* {*N*. *subprob-space N*}]
by(*auto intro*!: *arg-cong*[**where** *f*=λ*x*. *sets* (*borel-of x*)] *simp*: *Collect-conj-eq*[*symmetric*] *subprob-space-def*)
also have ... = *?rhs*
by(*auto simp*: *borel-of-subtopology sets-subprob-algebra-restrict*
weak-conv-topology-eq-finite-measure-algebra[*OF assms*]
intro!: *sets-restrict-space-cong*)
finally show *?thesis* **.**
qed

corollary *weak-conv-topology-eq-prob-algebra*:
assumes *Polish-space X*
shows *sets* (*borel-of* (*subtopology* (*weak-conv-topology X*) {*N*. *prob-space N* ∧ *sets N* = *sets* (*borel-of X*)}))
= *sets* (*prob-algebra* (*borel-of X*)) (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = *sets* (*borel-of* (*subtopology*
(*subtopology* (*weak-conv-topology X*) {*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of X*)})
{*N*. *prob-space N*}))
by(*auto simp*: *subtopology-subtopology Collect-conj-eq*[*symmetric*] *dest*:*prob-space-imp-subprob-space*
intro!: *arg-cong*[**where** *f*=λ*x*. *sets* (*borel-of* (*subtopology - x*))])
also have ... = *sets* (*restrict-space* (*borel-of* (*subtopology* (*weak-conv-topology X*)
{*N*. *subprob-space N* ∧ *sets N* = *sets* (*borel-of X*)})) {*N*. *prob-space N*})
by(*simp add*: *borel-of-subtopology*)
also have ... = *sets* (*restrict-space* (*subprob-algebra* (*borel-of X*)) {*N*. *prob-space N*})
by(*simp cong*: *sets-restrict-space-cong add*: *weak-conv-topology-eq-subprob-algebra*[*OF assms*])
also have ... = *?rhs*
by(*simp add*: *prob-algebra-def*)
finally show *?thesis* **.**
qed

6.3 Standardness

lemma *closedin-weak-conv-topology-r*:
closedin (*weak-conv-topology X*) {*N*. *sets N* = *sets* (*borel-of X*) ∧ *N* (*space N*) ≤ *ennreal r*}
proof(*rule closedin-limitin*)
fix *Ni N*
assume *h*:⋀*U*. *Ni U* ∈ *topspace* (*weak-conv-topology X*)
limitin (*weak-conv-topology X*) *Ni N* (*nhdsin-sets* (*weak-conv-topology X*) *N*)
⋀*U*. *N* ∈ *U* ⟹ *openin* (*weak-conv-topology X*) *U*
⟹ *Ni U* ∈ {*N*. *sets N* = *sets* (*borel-of X*) ∧ *emeasure N* (*space N*)

$\leq$ *ennreal r*}
have *x*: *sets N* = *sets* (*borel-of X*) *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **by** *auto*
interpret *N*: *finite-measure N*
by *fact*
interpret *Ni*: *finite-measure Ni i* **for** *i*
using *h*(*1*) **by** *simp*
have $\bigwedge f$. *continuous-map X euclideanreal f* $\Longrightarrow$ ($\exists$ *B*. $\forall$ *x*$\in$ *topspace X*. *abs* (*f x*) $\leq$ *B*)
$\Longrightarrow$ ((λn. $\int x$. *f x* ∂*Ni n*) $\longrightarrow$ ($\int x$. *f x* ∂N)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
using *h*(*2*) **by**(*auto simp*: *weak-conv-on-def*)
from *this*[*of* λx. *1*]
have ((λn. *measure* (*Ni n*) (*space* (*Ni n*))) $\longrightarrow$ *measure N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
by *auto*
hence ((λn. *Ni n* (*space* (*Ni n*))) $\longrightarrow$ *N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
by (*simp add*: *N*.*emeasure-eq-measure Ni*.*emeasure-eq-measure*)
hence *emeasure N* (*space N*) $\leq$ *ennreal r*
using *limitin-topspace*[*OF h*(*2*)] *h*(*3*) **by**(*auto intro*!: *tendsto-upperbound eventually-nhdsin-setsI*)
thus *N* $\in$ {*N*. *sets N* = *sets* (*borel-of X*) $\wedge$ *emeasure N* (*space N*) $\leq$ *ennreal r*}
using *x* **by** *blast*
qed (*auto intro*!: *finite-measureI simp*: *top.extremum-unique*)

lemma *closedin-weak-conv-topology-subprob*:
closedin (*weak-conv-topology X*) {*N*. *subprob-space N* $\wedge$ *sets N* = *sets* (*borel-of X*)}
proof(*rule closedin-limitin*)
fix *Ni N*
assume *h*:$\bigwedge U$. *Ni U* $\in$ *topspace* (*weak-conv-topology X*)
limitin (*weak-conv-topology X*) *Ni N* (*nhdsin-sets* (*weak-conv-topology X*) *N*)
$\bigwedge U$. *N* $\in$ *U* $\Longrightarrow$ *openin* (*weak-conv-topology X*) *U*
$\Longrightarrow$ *Ni U* $\in$ {*N*. *subprob-space N* $\wedge$ *sets N* = *sets* (*borel-of X*)}
have *x*: *sets N* = *sets* (*borel-of X*) *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **by** *auto*
have *X*:*topspace X* $\neq$ {}
using *h*(*3*)[*OF limitin-topspace*[*OF h*(*2*)],*simplified openin-topspace*]
by(*auto simp*: *subprob-space-def space-borel-of subprob-space-axioms-def cong*: *sets-eq-imp-space-eq*)
interpret *N*: *finite-measure N*
by *fact*
interpret *Ni*: *finite-measure Ni i* **for** *i*
using *h*(*1*) **by** *simp*
have $\bigwedge f$. *continuous-map X euclideanreal f* $\Longrightarrow$ ($\exists$ *B*. $\forall$ *x*$\in$ *topspace X*. *abs* (*f x*) $\leq$ *B*)
$\Longrightarrow$ ((λn. $\int x$. *f x* ∂*Ni n*) $\longrightarrow$ ($\int x$. *f x* ∂N)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)

using *h* **by**(*auto simp*: *weak-conv-on-def*)
from *this*[*of* λx. *1*]
have ((λn. *measure* (*Ni n*) (*space* (*Ni n*))) $\longrightarrow$ *measure N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
by *auto*
hence *1*:((λn. *Ni n* (*space* (*Ni n*))) $\longrightarrow$ *N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
by (*simp add*: *N.emeasure-eq-measure Ni.emeasure-eq-measure*)
hence *emeasure N* (*space N*) $\leq$ *1*
using *limitin-topspace*[*OF h*(*2*)] *h*(*3*)
by(*auto intro*!: *tendsto-upperbound*[*OF 1*] *eventually-nhdsin-setsI dest*:*subprob-space.subprob-emeasure-le-1*)
hence *subprob-space N*
using *X* **by**(*auto intro*!: *subprob-spaceI simp*: *sets-eq-imp-space-eq*[*OF x*(*1*)] *space-borel-of*)
thus *N* $\in$ {*N*. *subprob-space N* $\wedge$ *sets N* = *sets* (*borel-of X*)}
using *x h*(*3*) **by** *fast*
qed (*auto simp*: *subprob-space-def*)

lemma *closedin-weak-conv-topology-prob*:
closedin (*weak-conv-topology X*) {*N*. *prob-space N* $\wedge$ *sets N* = *sets* (*borel-of X*)}
proof(*rule closedin-limitin*)
fix *Ni N*
assume *h*:$\bigwedge$*U*. *Ni U* $\in$ *topspace* (*weak-conv-topology X*)
limitin (*weak-conv-topology X*) *Ni N* (*nhdsin-sets* (*weak-conv-topology X*) *N*)
$\bigwedge$*U*. *N* $\in$ *U* $\Longrightarrow$ *openin* (*weak-conv-topology X*) *U*
$\Longrightarrow$ *Ni U* $\in$ {*N*. *prob-space N* $\wedge$ *sets N* = *sets* (*borel-of X*)}
have *x*: *sets N* = *sets* (*borel-of X*) *finite-measure N*
using *limitin-topspace*[*OF h*(*2*)] **by** *auto*
interpret *N*: *finite-measure N*
by *fact*
interpret *Ni*: *finite-measure Ni i* **for** *i*
using *h*(*1*) **by** *simp*
have $\bigwedge$*f*. *continuous-map X euclideanreal f* $\Longrightarrow$ ($\exists$ *B*. $\forall$ *x*$\in$ *topspace X*. *abs* (*f x*) $\leq$ *B*)
$\Longrightarrow$ ((λn. $\int x$. *f x* ∂*Ni n*) $\longrightarrow$ ($\int x$. *f x* ∂*N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
using *h* **by**(*auto simp*: *weak-conv-on-def*)
from *this*[*of* λx. *1*]
have ((λn. *measure* (*Ni n*) (*space* (*Ni n*))) $\longrightarrow$ *measure N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
by *auto*
hence ((λn. *1*) $\longrightarrow$ *measure N* (*space N*)) (*nhdsin-sets* (*weak-conv-topology X*) *N*)
using *x h*
by(*auto intro*!: *tendsto-cong*[**where** *f*=λn. *measure* (*Ni n*) (*space* (*Ni n*))
and *g*=λn. *1*,*THEN iffD1*] *eventually-nhdsin-setsI prob-space.prob-space*)
hence *measure N* (*space N*) = *1*
by (*metis nhdsin-sets-bot h*(*2*) *limitin-topspace tendsto-const-iff*)
hence *prob-space N*

```
    by (simp add: N.emeasure-eq-measure prob-spaceI)
  thus N ∈ {N. prob-space N ∧ sets N = sets (borel-of X)}
    using x by blast
qed (auto simp: prob-space.finite-measure)


corollary
  assumes standard-borel M
  shows standard-borel-finite-measure-algebra: standard-borel (finite-measure-algebra
M)
    and standard-borel-ne-finite-measure-algebra: standard-borel-ne (finite-measure-algebra
M)
    and standard-borel-subprob-algebra: standard-borel (subprob-algebra M)
    and standard-borel-prob-algebra: standard-borel (prob-algebra M)
proof −
  interpret sbn: standard-borel M by fact
  obtain X where X: Polish-space X sets M = sets (borel-of X)
    using sbn.Polish-space by blast
  show 1:standard-borel (finite-measure-algebra M)
    by (metis X finite-measure-algebra-cong Polish-space-weak-conv-topology stan-
dard-borel.intro weak-conv-topology-eq-finite-measure-algebra)
  moreover have null-measure M ∈ space (finite-measure-algebra M)
    by(auto simp: space-finite-measure-algebra intro!: finite-measureI)
  ultimately show standard-borel-ne (finite-measure-algebra M)
    using standard-borel-ne-axioms-def standard-borel-ne-def by force
  show standard-borel (subprob-algebra M)
    using Polish-space-closedin[OF Polish-space-weak-conv-topology[OF X(1)] closedin-weak-conv-topology-subp

    by(auto cong: subprob-algebra-cong
            simp: X(2) weak-conv-topology-eq-subprob-algebra[OF X(1),symmetric]
standard-borel-def)
  show standard-borel (prob-algebra M)
    using Polish-space-closedin[OF Polish-space-weak-conv-topology[OF X(1)] closedin-weak-conv-topology-prob

    by(auto cong: prob-algebra-cong
              simp: X(2) weak-conv-topology-eq-prob-algebra[OF X(1),symmetric]
standard-borel-def)
qed

corollary
  assumes standard-borel-ne M
  shows standard-borel-ne-subprob-algebra: standard-borel-ne (subprob-algebra M)
    and standard-borel-ne-prob-algebra: standard-borel-ne (prob-algebra M)
proof −
  obtain x where x: x ∈ space M
    using assms standard-borel-ne.space-ne by auto
  then have return M x ∈ space (subprob-algebra M) return M x ∈ space (prob-algebra
M)
    using prob-space-return
```

by(*auto intro*!: *prob-space-imp-subprob-space simp*: *space-subprob-algebra space-prob-algebra*)
thus *standard-borel-ne* (*subprob-algebra M*) *standard-borel-ne* (*prob-algebra M*)
using *assms standard-borel-subprob-algebra standard-borel-prob-algebra*
by(*auto simp*: *standard-borel-ne-def standard-borel-ne-axioms-def*)
qed

end

References

[1] C. E. Heil. Lecture note on math 6338 (real analysis ii) at Georgia Institute of Technology. https://heil.math.gatech.edu/6338/summer08/, 2008. Accessed November 17th 2023.

[2] A. S. Kechris. *Classical Descriptive Set Theory*. Graduate Texts in Mathematics. Springer New York, 1995.

[3] M. Shi. Nets and filters. https://www.uvm.edu/~smillere/TProjects/MShi20s.pdf, 2020. Accessed November 17th 2023.

[4] O. van Gaans. Probability measures on metric spaces. https://www.math.leidenuniv.nl/~vangaans/jancol1.pdf. Accessed: March 2. 2023.