Low Degree Hypergraphs

Ata Keskin

July 20, 2023

Abstract

Martingale poopies: [1], I love my bebis alara!

Contents

1 Introduction

Alara is best bebis ever!
theory *Measure-Space-Addendum*
imports *HOL−Analysis.Measure-Space*
begin

1.1 Sigma Algebra Generated by a Family of Functions

definition *sigma-gen* :: $'a\ set \Rightarrow 'b\ measure \Rightarrow ('a \Rightarrow 'b)\ set \Rightarrow 'a\ measure$ **where**
$sigma\text{-}gen\ \Omega\ N\ S \equiv sigma\ \Omega\ (\bigcup f \in S.\ \{f\ -`\ A \cap \Omega \mid A.\ A \in N\})$

lemma [*simp*]:
shows *sets-sigma-gen*: $sets\ (sigma\text{-}gen\ \Omega\ N\ S) = sigma\text{-}sets\ \Omega\ (\bigcup f \in S.\ \{f\ -`\ A \cap \Omega \mid A.\ A \in N\})$
and *space-sigma-gen*: $space\ (sigma\text{-}gen\ \Omega\ N\ S) = \Omega$
by (*auto simp add*: *sigma-gen-def sets-measure-of-conv space-measure-of-conv*)

lemma *measurable-sigma-gen*:
assumes $f \in S$ $f \in \Omega \to space\ N$
shows $f \in sigma\text{-}gen\ \Omega\ N\ S \to_M N$
using *assms* **by** (*intro measurableI*, *auto*)

lemma *measurable-sigma-gen-singleton*:
assumes $f \in \Omega \to space\ N$
shows $f \in sigma\text{-}gen\ \Omega\ N\ \{f\} \to_M N$
using *assms measurable-sigma-gen* **by** *blast*

lemma *measurable-iff-contains-sigma-gen*:
shows $(f \in M \to_M N) \longleftrightarrow f \in space\ M \to space\ N \wedge sigma\text{-}gen\ (space\ M)\ N\ \{f\} \subseteq M$
proof (*standard*, *goal-cases*)
case *1*
hence $f \in space\ M \to space\ N$ **using** *measurable-space* **by** *fast*
thus *?case* **unfolding** *sets-sigma-gen* **by** (*simp*, *intro sigma-algebra.sigma-sets-subset*, (*blast intro*: *sets.sigma-algebra-axioms measurable-sets*[*OF 1*])+)
next
case *2*
thus *?case* **using** *measurable-mono*[*OF - refl - space-sigma-gen*, *of N M*] *measurable-sigma-gen-singleton* **by** *fast*
qed

lemma *measurable-iff-contains-sigma-gen'*:
shows $(S \subseteq M \to_M N) \longleftrightarrow S \subseteq space\ M \to space\ N \wedge sigma\text{-}gen\ (space\ M)\ N\ S \subseteq M$
proof (*standard*, *goal-cases*)
case *1*
hence *subset*: $S \subseteq space\ M \to space\ N$ **using** *measurable-space* **by** *fast*
have $\{f\ -`\ A \cap space\ M \mid A.\ A \in N\} \subseteq M$ **if** $f \in S$ **for** f **using** *measur-*

able-iff-contains-sigma-gen[*unfolded sets-sigma-gen*, *of f*] *1 subset that* **by** *blast*
then show *?case* **unfolding** *sets-sigma-gen* **using** *sets.sigma-algebra-axioms* **by** (*simp add*: *subset*, *intro sigma-algebra.sigma-sets-subset*, *blast*+)
next
case *2*
hence *subset*: $S \subseteq$ *space M* $\rightarrow$ *space N* **by** *simp*
show *?case*
proof (*standard*, *goal-cases*)
case (*1 x*)
have *sigma-gen* (*space M*) *N* $\{x\} \subseteq M$ **by** (*metis* (*no-types*, *lifting*) *1 2 sets-sigma-gen SUP-le-iff sigma-sets-le-sets-iff singletonD*)
thus *?case* **using** *measurable-iff-contains-sigma-gen subset*[*THEN subsetD*, *OF 1*] **by** *fast*
qed
qed

end
theory *Elementary-Metric-Spaces-Addendum*
imports *HOL−Analysis.Elementary-Metric-Spaces HOL−Analysis.Bochner-Integration*
begin

lemma *diameter-comp-strict-mono*:
fixes *s* :: *nat* $\Rightarrow$ *'a* :: *real-normed-vector*
assumes *strict-mono r bounded* $\{s\ i\ | i.\ r\ n \le i\}$
shows *diameter* $\{s\ (r\ i)\ |\ i.\ n \le i\} \le$ *diameter* $\{s\ i\ |\ i.\ r\ n \le i\}$
proof (*rule diameter-subset*)
show $\{s\ (r\ i)\ |\ i.\ n \le i\} \subseteq \{s\ i\ |\ i.\ r\ n \le i\}$ **using** *assms*(*1*) *monotoneD strict-mono-mono* **by** *fastforce*
qed (*intro assms*(*2*))

lemma *diameter-bounded-bound'*:
fixes *S* :: *'a* :: *metric-space set*
assumes *S*: *bdd-above* (*case-prod dist* ‘ ($S \times S$)) $x \in S$ $y \in S$
shows *dist x y* $\le$ *diameter S*
proof −
have $(x,y) \in S \times S$ **using** *S* **by** *auto*
then have *dist x y* $\le$ (*SUP* $(x,y) \in S \times S$. *dist x y*) **by** (*rule cSUP-upper2*[*OF assms*(*1*)]) *simp*
with ‹$x \in S$› **show** *?thesis* **by** (*auto simp*: *diameter-def*)
qed

lemma *bounded-imp-dist-bounded*:
assumes *bounded* (*range s*)
shows *bounded* (($\lambda(i, j)$. *dist* (*s i*) (*s j*)) ‘ ($\{n..\} \times \{n..\}$))
using *bounded-dist-comp*[*OF bounded-fst bounded-snd*, *OF bounded-Times*(*1*,*1*)[*OF assms*(*1*,*1*)]] **by** (*rule bounded-subset*, *force*)

lemma *cauchy-iff-diameter-tends-to-zero-and-bounded*:
fixes *s* :: *nat* $\Rightarrow$ *'a* :: *real-normed-vector*

```
  shows Cauchy s ⟷ ((λn. diameter {s i | i. i ≥ n}) ⟶ 0 ∧ bounded (range
s))
proof −
  have {s i |i. N ≤ i} ≠ {} for N by blast
  hence diameter-SUP: diameter {s i |i. N ≤ i} = (SUP (i, j) ∈ {N..} × {N..}.
dist (s i) (s j)) for N unfolding diameter-def by (auto intro!: arg-cong[of - - Sup])
  show ?thesis
  proof ((standard ; clarsimp), goal-cases)
    case 1
    have ∃ N. ∀ n≥N. norm (diameter {s i |i. n ≤ i}) < e if e-pos: e > 0 for e
    proof −
      obtain N where dist-less: dist (s n) (s m) < (1/2) ∗ e if n ≥ N m ≥ N for
n m using 1 CauchyD e-pos dist-norm by (metis mult-pos-pos zero-less-divide-iff
zero-less-numeral zero-less-one)
      {
        fix r assume r ≥ N
        hence dist (s n) (s m) < (1/2) ∗ e if n ≥ r m ≥ r for n m using dist-less
that by simp
         hence (SUP (i, j) ∈ {r..} × {r..}. dist (s i) (s j)) ≤ (1/2) ∗ e by (intro
cSup-least) fastforce+
        also have ... < e using e-pos by simp
       finally have diameter {s i |i. r ≤ i} < e using diameter-SUP by presburger
      }
      moreover have diameter {s i |i. r ≤ i} ≥ 0 for r unfolding diameter-SUP
using bounded-imp-dist-bounded[OF cauchy-imp-bounded, THEN bounded-imp-bdd-above,
OF 1] by (intro cSup-upper2, auto)
      ultimately show ?thesis by auto
    qed
    thus ?case using cauchy-imp-bounded[OF 1] by (simp add: LIMSEQ-iff)
  next
    case 2
    have ∃ N. ∀ n≥N. ∀ m≥N. dist (s n) (s m) < e if e-pos: e > 0 for e
    proof −
       obtain N where diam-less: diameter {s i |i. r ≤ i} < e if r ≥ N for r
using LIMSEQ-D 2(1) e-pos by fastforce
      {
        fix n m assume n ≥ N m ≥ N
      hence dist (s n) (s m) < e using cSUP-lessD[OF bounded-imp-dist-bounded[THEN
bounded-imp-bdd-above], OF 2(2) diam-less[unfolded diameter-SUP]] by auto
      }
      thus ?thesis by blast
    qed
    then show ?case by (intro CauchyI, simp add: dist-norm)
  qed
qed

context
  fixes s r :: nat ⇒ 'a ⇒ 'b :: {second-countable-topology, real-normed-vector,
banach} and M
```

assumes *bounded*: $\bigwedge x.\ x \in space\ M \Longrightarrow bounded\ (range\ (\lambda i.\ s\ i\ x))$
begin

lemma *borel-measurable-diameter*:
assumes [*measurable*]: $\bigwedge i.\ (s\ i) \in borel\text{-}measurable\ M$
shows $(\lambda x.\ diameter\ \{s\ i\ x\ |i.\ n \leq i\}) \in borel\text{-}measurable\ M$
proof $-$
have $\{s\ i\ x\ |i.\ N \leq i\} \neq \{\}$ **for** $x\ N$ **by** *blast*
hence *diameter-SUP*: $diameter\ \{s\ i\ x\ |i.\ N \leq i\} = (SUP\ (i, j) \in \{N..\} \times \{N..\}.\ dist\ (s\ i\ x)\ (s\ j\ x))$ **for** $x\ N$ **unfolding** *diameter-def* **by** (*auto intro*!: *arg-cong*[*of* - - *Sup*])

have $case\text{-}prod\ dist\ `\ (\{s\ i\ x\ |i.\ n \leq i\} \times \{s\ i\ x\ |i.\ n \leq i\}) = ((\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\}))$ **for** x **by** *fast*
hence $*$: $(\lambda x.\ diameter\ \{s\ i\ x\ |i.\ n \leq i\}) = (\lambda x.\ Sup\ ((\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\})))$ **using** *diameter-SUP* **by** (*simp add*: *case-prod-beta'*)

have $bounded\ ((\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\}))$ **if** $x \in space\ M$ **for** x **by** (*rule bounded-imp-dist-bounded*[*OF bounded*, *OF that*])
hence *bdd*: $bdd\text{-}above\ ((\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\}))$ **if** $x \in space\ M$ **for** x **using** *that bounded-imp-bdd-above* **by** *presburger*
have $fst\ p \in borel\text{-}measurable\ M\ snd\ p \in borel\text{-}measurable\ M$ **if** $p \in s\ `\ \{n..\} \times s\ `\ \{n..\}$ **for** p **using** *that* **by** *fastforce+*
hence $(\lambda x.\ fst\ p\ x - snd\ p\ x) \in borel\text{-}measurable\ M$ **if** $p \in s\ `\ \{n..\} \times s\ `\ \{n..\}$ **for** p **using** *that borel-measurable-diff* **by** *simp*
hence $(\lambda x.\ case\ p\ of\ (f, g) \Rightarrow dist\ (f\ x)\ (g\ x)) \in borel\text{-}measurable\ M$ **if** $p \in s\ `\ \{n..\} \times s\ `\ \{n..\}$ **for** p **unfolding** *dist-norm* **using** *that* **by** *measurable*
moreover have $countable\ (s\ `\ \{n..\} \times s\ `\ \{n..\})$ **by** (*intro countable-SIGMA countable-image*, *auto*)
ultimately show *?thesis* **unfolding** $*$ **by** (*auto intro*!: *borel-measurable-cSUP bdd*)
qed

lemma *integrable-bound-diameter*:
fixes $f :: 'a \Rightarrow real$
assumes *integrable M f*
and [*measurable*]: $\bigwedge i.\ (s\ i) \in borel\text{-}measurable\ M$
and $\bigwedge x\ i.\ x \in space\ M \Longrightarrow norm\ (s\ i\ x) \leq f\ x$
shows $integrable\ M\ (\lambda x.\ diameter\ \{s\ i\ x\ |i.\ n \leq i\})$
proof $-$
have $\{s\ i\ x\ |i.\ N \leq i\} \neq \{\}$ **for** $x\ N$ **by** *blast*
hence *diameter-SUP*: $diameter\ \{s\ i\ x\ |i.\ N \leq i\} = (SUP\ (i, j) \in \{N..\} \times \{N..\}.\ dist\ (s\ i\ x)\ (s\ j\ x))$ **for** $x\ N$ **unfolding** *diameter-def* **by** (*auto intro*!: *arg-cong*[*of* - - *Sup*])
{
fix x **assume** x: $x \in space\ M$
let $?S = (\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\})$
have $case\text{-}prod\ dist\ `\ (\{s\ i\ x\ |i.\ n \leq i\} \times \{s\ i\ x\ |i.\ n \leq i\}) = (\lambda(i, j).\ dist\ (s\ i\ x)\ (s\ j\ x))\ `\ (\{n..\} \times \{n..\})$ **by** *fast*

hence $*$: *diameter* $\{s\ i\ x\ |i.\ n \le i\} = $ *Sup ?S* **using** *diameter-SUP* **by** (*simp add*: *case-prod-beta'*)

have *bounded ?S* **by** (*rule bounded-imp-dist-bounded*[*OF bounded*[*OF x*]])
hence *Sup-S-nonneg*:$0 \le$ *Sup ?S* **by** (*auto intro*!: *cSup-upper2 x bounded-imp-bdd-above*)

have *dist* $(s\ i\ x)\ (s\ j\ x) \le\ 2 * f\ x$ **for** $i\ j$ **by** (*intro dist-triangle2*[*THEN order-trans*, *of - 0*]) (*metis norm-conv-dist assms*(*3*) *x add-mono mult-2*)
hence $\forall\, c \in$ *?S*. $c \le 2 * f\ x$ **by** *force*
hence *Sup ?S* $\le 2 * f\ x$ **by** (*intro cSup-least*, *auto*)
hence *norm* (*Sup ?S*) $\le 2 *$ *norm* $(f\ x)$ **using** *Sup-S-nonneg* **by** *auto*
also have ... = *norm* $(2 *_R f\ x)$ **by** *simp*
finally have *norm* (*diameter* $\{s\ i\ x\ |i.\ n \le i\}) \le$ *norm* $(2 *_R f\ x)$ **unfolding** $*$ **.**
}
hence *AE x in M*. *norm* (*diameter* $\{s\ i\ x\ |i.\ n \le i\}) \le$ *norm* $(2 *_R f\ x)$ **by** *blast*
thus *integrable M* ($\lambda x.$ *diameter* $\{s\ i\ x\ |i.\ n \le i\}$) **using** *borel-measurable-diameter* **by** (*intro Bochner-Integration.integrable-bound*[*OF assms*(*1*)[*THEN integrable-scaleR-right*[*of 2*]]], *measurable*)
qed
end

end
theory *Bochner-Integration-Addendum*
imports *HOL−Analysis.Bochner-Integration*
begin

1.2 Simple Functions

lemma *integrable-implies-simple-function-sequence*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes *integrable M f*
obtains s **where** $\bigwedge i.$ *simple-function M* $(s\ i)$
and $\bigwedge i.$ *emeasure M* $\{y \in$ *space M*. $s\ i\ y \neq 0\} \neq \infty$
and $\bigwedge x.\ x \in$ *space M* $\Longrightarrow (\lambda i.\ s\ i\ x) \longrightarrow f\ x$
and $\bigwedge x\ i.\ x \in$ *space M* $\Longrightarrow$ *norm* $(s\ i\ x) \le 2 *$ *norm* $(f\ x)$
proof−
have f: $f \in$ *borel-measurable M* $(\int^+ x.$ *norm* $(f\ x)\ \partial M) < \infty$ **using** *assms* **unfolding** *integrable-iff-bounded* **by** *auto*
obtain s **where** s: $\bigwedge i.$ *simple-function M* $(s\ i)$ $\bigwedge x.\ x \in$ *space M* $\Longrightarrow (\lambda i.\ s\ i\ x) \longrightarrow f\ x$ $\bigwedge i\ x.\ x \in$ *space M* $\Longrightarrow$ *norm* $(s\ i\ x) \le 2 *$ *norm* $(f\ x)$ **using** *borel-measurable-implies-sequence-metric*[*OF f*(*1*)] **unfolding** *norm-conv-dist* **by** *metis*
{
fix i
have $(\int^+ x.$ *norm* $(s\ i\ x)\ \partial M) \le (\int^+ x.$ *ennreal* $(2 *$ *norm* $(f\ x))\ \partial M)$ **using** s **by** (*intro nn-integral-mono*, *auto*)
also have $\ldots < \infty$ **using** f **by** (*simp add*: *nn-integral-cmult ennreal-mult-less-top ennreal-mult*)

finally have *sbi*: *Bochner-Integration.simple-bochner-integrable M* (*s i*) **using** *s* **by** (*intro simple-bochner-integrableI-bounded*) *auto*
hence *emeasure M* $\{y \in$ *space M*. *s i y* $\neq 0\} \neq \infty$ **by** (*auto intro*: *integrableI-simple-bochner-integrable simple-bochner-integrable.cases*)
}
thus *?thesis* **using** *that s* **by** *blast*
qed

lemma *banach-simple-function-indicator-representation*:
fixes *f* ::$'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*}
assumes *f*: *simple-function M f* **and** *x*: $x \in$ *space M*
shows $f\ x = (\sum y \in f$ ‘ *space M*. *indicator* (f −‘ $\{y\} \cap$ *space M*) $x *_R y$)
(**is** *?l* = *?r*)
proof −
have *?r* = ($\sum y \in f$ ‘ *space M*.
(*if* $y = f\ x$ *then indicator* (f −‘ $\{y\} \cap$ *space M*) $x *_R y$ *else 0*)) **by** (*auto intro*!: *sum.cong*)
also have ... = *indicator* (f −‘ $\{f\ x\} \cap$ *space M*) $x *_R f\ x$ **using** *assms* **by** (*auto dest*: *simple-functionD*)
also have ... = $f\ x$ **using** *x* **by** (*auto simp*: *indicator-def*)
finally show *?thesis* **by** *auto*
qed

lemma *banach-simple-function-indicator-representation-AE*:
fixes *f* ::$'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*}
assumes *f*: *simple-function M f*
shows *AE x in M*. $f\ x = (\sum y \in f$ ‘ *space M*. *indicator* (f −‘ $\{y\} \cap$ *space M*) $x *_R y$)
by (*metis* (*mono-tags*, *lifting*) *AE-I2 banach-simple-function-indicator-representation f*)

lemmas *simple-function-scaleR*[*intro*] = *simple-function-compose2*[**where** h=($*_R$)]

lemma *integrable-simple-function*:
assumes *simple-function M f emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty$
shows *integrable M f*
using *assms has-bochner-integral-simple-bochner-integrable integrable.simps simple-bochner-integrable.simps* **by** *blast*

lemma *simple-integrable-function-induct*[*consumes 2*, *case-names cong indicator add*, *induct set*: *simple-function*]:
fixes *f* :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*}
assumes *f*: *simple-function M f emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty$
assumes *cong*: $\bigwedge f\ g$. *simple-function M f* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty \Longrightarrow$ *simple-function M g* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $g\ y \neq 0\} \neq \infty \Longrightarrow (\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x = g\ x) \Longrightarrow P\ f \Longrightarrow P\ g$
assumes *indicator*: $\bigwedge A\ y.\ A \in$ *sets M* $\Longrightarrow$ *emeasure M A* $< \infty \Longrightarrow P$ (λx. *indicator A x* $*_R y$)
assumes *add*: $\bigwedge f\ g$. *simple-function M f* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $f\ y \neq$

$0\} \neq \infty \Longrightarrow$

simple-function M $g \Longrightarrow$ *emeasure* M $\{y \in$ *space* $M.$ g $y \neq 0\} \neq$
$\infty \Longrightarrow$

$(\bigwedge z.$ $z \in$ *space* $M \Longrightarrow$ *norm* $(f$ z $+$ g $z)$ $=$ *norm* $(f$ $z)$ $+$ *norm*
$(g$ $z)) \Longrightarrow$

P $f \Longrightarrow P$ $g \Longrightarrow P$ $(\lambda x.$ f x $+$ g $x)$

shows P f
proof−
let *?f* $= \lambda x.$ $(\sum y \in f$ ‘ *space* $M.$ *indicat-real* $(f$ −‘ $\{y\} \cap$ *space* $M)$ $x *_R y)$
have *f-ae-eq*: f x $=$ *?f* x **if** $x \in$ *space* M **for** x **using** *banach-simple-function-indicator-representation*[*OF* $f(1)$ *that*] **.**
moreover have *emeasure* M $\{y \in$ *space* $M.$ *?f* $y \neq 0\} \neq \infty$ **by** (*metis* (*no-types*, *lifting*) *Collect-cong calculation* $f(2)$)
moreover have P $(\lambda x.$ $\sum y \in S.$ *indicat-real* $(f$ −‘ $\{y\} \cap$ *space* $M)$ $x *_R y)$
simple-function M $(\lambda x.$ $\sum y \in S.$ *indicat-real* $(f$ −‘ $\{y\} \cap$ *space* $M)$ x $*_R y)$
emeasure M $\{y \in$ *space* $M.$ $(\sum x \in S.$ *indicat-real* $(f$ −‘ $\{x\} \cap$ *space* $M)$ $y *_R x) \neq 0\} \neq \infty$
if $S \subseteq f$ ‘ *space* M **for** S **using** *simple-functionD*(*1*)[*OF assms*(*1*), *THEN rev-finite-subset*, *OF that*] *that*
proof (*induction rule*: *finite-induct*)
case *empty*
{
case *1*
then show *?case* **using** *indicator*[*of* {}] **by** *force*
next
case *2*
then show *?case* **by** *force*
next
case *3*
then show *?case* **by** *force*
}
next
case (*insert x F*)
have $(f$ −‘ $\{x\} \cap$ *space* $M) \subseteq \{y \in$ *space* $M.$ f $y \neq 0\}$ **if** $x \neq 0$ **using** *that* **by** *blast*
moreover have $\{y \in$ *space* $M.$ f $y \neq 0\}$ $=$ *space* M $-$ $(f$ −‘ $\{0\} \cap$ *space* $M)$ **by** *blast*
moreover have *space* M $-$ $(f$ −‘ $\{0\} \cap$ *space* $M) \in$ *sets* M **using** *simple-functionD*(*2*)[*OF* $f(1)$] **by** *blast*
ultimately have *fin-0*: *emeasure* M $(f$ −‘ $\{x\} \cap$ *space* $M) < \infty$ **if** $x \neq 0$ **using** *that* **by** (*metis emeasure-mono* $f(2)$ *infinity-ennreal-def top.not-eq-extremum top-unique*)
hence *fin-1*: *emeasure* M $\{y \in$ *space* $M.$ *indicat-real* $(f$ −‘ $\{x\} \cap$ *space* $M)$ $y *_R x \neq 0\} \neq \infty$ **if** $x \neq 0$ **by** (*metis* (*mono-tags*, *lifting*) *emeasure-mono* $f(1)$ *indicator-simps*(*2*) *linorder-not-less mem-Collect-eq scaleR-eq-0-iff simple-functionD*(*2*) *subsetI that*)

have $*$: $(\sum y \in$*insert* x $F.$ *indicat-real* $(f$ −‘ $\{y\} \cap$ *space* $M)$ $xa *_R y) = (\sum y \in F.$

indicat-real $(f -\text{`} \{y\} \cap$ *space M*$)$ *xa* $*_R$ *y*$)$ + *indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *xa* $*_R$ *x* **for** *xa* **by** (*metis* (*no-types*, *lifting*) *Diff-empty Diff-insert0 add.commute insert.hyps*(*1*) *insert.hyps*(*2*) *sum.insert-remove*)

have $**$: $\{y \in$ *space M*. $(\sum x{\in}$*insert x F*. *indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *y* $*_R$ *x*$) \neq 0\} \subseteq \{y \in$ *space M*. $(\sum x{\in}F$. *indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *y* $*_R$ *x*$) \neq 0\} \cup \{y \in$ *space M*. *indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *y* $*_R$ *x* $\neq 0\}$ **unfolding** $*$ **by** *fastforce*

{

case *1*

hence *x*: $x \in f$ ` *space M* **and** *F*: $F \subseteq f$ ` *space M* **by** *auto*

show *?case*

proof (*cases* $x = 0$)

case *True*

then show *?thesis* **unfolding** $*$ **using** *insert*(*3*)[*OF F*] **by** *simp*

next

case *False*

have *norm-argument*: *norm* $((\sum y{\in}F$. *indicat-real* $(f -\text{`} \{y\} \cap$ *space M*$)$ *z* $*_R$ *y*$)$ + *indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *z* $*_R$ *x*$)$ = *norm* $(\sum y{\in}F$. *indicat-real* $(f -\text{`} \{y\} \cap$ *space M*$)$ *z* $*_R$ *y*$)$ + *norm* (*indicat-real* $(f -\text{`} \{x\} \cap$ *space M*$)$ *z* $*_R$ *x*$)$ **if** *z*: $z \in$ *space M* **for** *z*

proof (*cases f z* $= x$)

case *True*

have *indicat-real* $(f -\text{`} \{y\} \cap$ *space M*$)$ *z* $*_R$ *y* $= 0$ **if** $y \in F$ **for** *y* **using** *True insert*(*2*) *z that 1* **unfolding** *indicator-def* **by** *force*

hence $(\sum y{\in}F$. *indicat-real* $(f -\text{`} \{y\} \cap$ *space M*$)$ *z* $*_R$ *y*$) = 0$ **by** (*meson sum.neutral*)

then show *?thesis* **by** *force*

next

case *False*

then show *?thesis* **by** *force*

qed

show *?thesis* **using** *False simple-functionD*(*2*)[*OF f*(*1*)] *insert*(*3*,*5*)[*OF F*] *simple-function-scaleR fin-0 fin-1* **by** (*subst* $*$, *subst add*, *subst simple-function-sum*) (*blast intro*: *norm-argument indicator*)+

qed

next

case *2*

hence *x*: $x \in f$ ` *space M* **and** *F*: $F \subseteq f$ ` *space M* **by** *auto*

show *?case*

proof (*cases* $x = 0$)

case *True*

then show *?thesis* **unfolding** $*$ **using** *insert*(*4*)[*OF F*] **by** *simp*

next

case *False*

then show *?thesis* **unfolding** $*$ **using** *insert*(*4*)[*OF F*] *simple-functionD*(*2*)[*OF f*(*1*)] **by** *fast*

qed

next

case *3*

hence *x*: $x \in f$ ‘ *space M* **and** *F*: $F \subseteq f$ ‘ *space M* **by** *auto*
show *?case*
proof (*cases* $x = 0$)
case *True*
then show *?thesis* **unfolding** $*$ **using** *insert*(*5*)[*OF F*] **by** *simp*
next
case *False*
have *emeasure M* $\{y \in$ *space M*. ($\sum x\in$*insert x F*. *indicat-real* (f −‘ $\{x\}$ $\cap$ *space M*) $y *_R x) \neq 0\} \leq$ *emeasure M* ($\{y \in$ *space M*. ($\sum x\in F$. *indicat-real* (f −‘ $\{x\} \cap$ *space M*) $y *_R x) \neq 0\} \cup \{y \in$ *space M*. *indicat-real* (f −‘ $\{x\} \cap$ *space M*) $y *_R x \neq 0\}$)
using $**$ *simple-functionD*(*2*)[*OF insert*(*4*)[*OF F*]] *simple-functionD*(*2*)[*OF f*(*1*)] **by** (*intro emeasure-mono*, *force*+)
also have ... $\leq$ *emeasure M* $\{y \in$ *space M*. ($\sum x\in F$. *indicat-real* (f −‘ $\{x\}$ $\cap$ *space M*) $y *_R x) \neq 0\}$ + *emeasure M* $\{y \in$ *space M*. *indicat-real* (f −‘ $\{x\} \cap$ *space M*) $y *_R x \neq 0\}$
using *simple-functionD*(*2*)[*OF insert*(*4*)[*OF F*]] *simple-functionD*(*2*)[*OF f*(*1*)] **by** (*intro emeasure-subadditive*, *force*+)
also have ... $< \infty$ **using** *insert*(*5*)[*OF F*] *fin-1*[*OF False*] **by** (*simp add*: *less-top*)
finally show *?thesis* **by** *simp*
qed
}
qed
moreover have *simple-function M* (λx. $\sum y\in f$ ‘ *space M*. *indicat-real* (f −‘ $\{y\}$ $\cap$ *space M*) $x *_R y$) **using** *calculation* **by** *blast*
moreover have *P* (λx. $\sum y\in f$ ‘ *space M*. *indicat-real* (f −‘ $\{y\} \cap$ *space M*) x $*_R y$) **using** *calculation* **by** *blast*
ultimately show *?thesis* **by** (*intro cong*[*OF - - f*(*1*,*2*)], *blast*, *presburger*+)
qed

lemma *simple-integrable-function-induct-nonneg*[*consumes 3*, *case-names cong indicator add*, *induct set*: *simple-function*]:
fixes *f* :: *'a* $\Rightarrow$ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *f*: *simple-function M f emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty$ $\bigwedge x$. $x \in$ *space M* $\longrightarrow f\ x \geq 0$
assumes *cong*: $\bigwedge f\ g$. *simple-function M f* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $f\ y$ $\neq 0\} \neq \infty \Longrightarrow (\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x \geq 0) \Longrightarrow$ *simple-function M g* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $g\ y \neq 0\} \neq \infty \Longrightarrow (\bigwedge x.\ x \in$ *space M* $\Longrightarrow g\ x \geq 0)$ $\Longrightarrow (\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x = g\ x) \Longrightarrow P\ f \Longrightarrow P\ g$
assumes *indicator*: $\bigwedge A\ y.\ y \geq 0 \Longrightarrow A \in$ *sets M* $\Longrightarrow$ *emeasure M A* $< \infty \Longrightarrow$ *P* (λx. *indicator A x* $*_R y$)
assumes *add*: $\bigwedge f\ g.\ (\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x \geq 0) \Longrightarrow$ *simple-function M f* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty \Longrightarrow$
$(\bigwedge x.\ x \in$ *space M* $\Longrightarrow g\ x \geq 0) \Longrightarrow$ *simple-function M g* $\Longrightarrow$ *emeasure M* $\{y \in$ *space M*. $g\ y \neq 0\} \neq \infty \Longrightarrow$
$(\bigwedge z.\ z \in$ *space M* $\Longrightarrow$ *norm* $(f\ z + g\ z)$ = *norm* $(f\ z)$ + *norm* $(g\ z)) \Longrightarrow$

$P\ f \Longrightarrow P\ g \Longrightarrow P\ (\lambda x.\ f\ x + g\ x)$
shows $P\ f$
proof−
let $?f = \lambda x.\ (\sum y \in f\ `\ space\ M.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ x *_R\ y)$
have *f-ae-eq*: $f\ x = ?f\ x$ **if** $x \in space\ M$ **for** x **using** *banach-simple-function-indicator-representation*[*OF f*(*1*) *that*] **.**
moreover have $emeasure\ M\ \{y \in space\ M.\ ?f\ y \neq 0\} \neq \infty$ **by** (*metis* (*no-types*, *lifting*) *Collect-cong calculation f*(*2*))
moreover have $P\ (\lambda x.\ \sum y \in S.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ x *_R\ y)$
$simple\text{-}function\ M\ (\lambda x.\ \sum y \in S.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ x *_R\ y)$
$emeasure\ M\ \{y \in space\ M.\ (\sum x \in S.\ indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ y *_R\ x) \neq 0\} \neq \infty$
$\bigwedge x.\ x \in space\ M \Longrightarrow 0 \leq (\sum y \in S.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ x *_R\ y)$
if $S \subseteq f\ `\ space\ M$ **for** S **using** *simple-functionD*(*1*)[*OF assms*(*1*), *THEN rev-finite-subset*, *OF that*] *that*
proof (*induction rule*: *finite-subset-induct′*)
case *empty*
{
case *1*
then show *?case* **using** *indicator*[*of 0* {}] **by** *force*
next
case *2*
then show *?case* **by** *force*
next
case *3*
then show *?case* **by** *force*
next
case *4*
then show *?case* **by** *force*
}
next
case (*insert x F*)
have $(f\ -`\ \{x\} \cap space\ M) \subseteq \{y \in space\ M.\ f\ y \neq 0\}$ **if** $x \neq 0$ **using** *that* **by** *blast*
moreover have $\{y \in space\ M.\ f\ y \neq 0\} = space\ M - (f\ -`\ \{0\} \cap space\ M)$ **by** *blast*
moreover have $space\ M - (f\ -`\ \{0\} \cap space\ M) \in sets\ M$ **using** *simple-functionD*(*2*)[*OF f*(*1*)] **by** *blast*
ultimately have *fin-0*: $emeasure\ M\ (f\ -`\ \{x\} \cap space\ M) < \infty$ **if** $x \neq 0$ **using** *that* **by** (*metis emeasure-mono f*(*2*) *infinity-ennreal-def top.not-eq-extremum top-unique*)
hence *fin-1*: $emeasure\ M\ \{y \in space\ M.\ indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ y *_R\ x \neq 0\} \neq \infty$ **if** $x \neq 0$ **by** (*metis* (*mono-tags*, *lifting*) *emeasure-mono f*(*1*) *indicator-simps*(*2*) *linorder-not-less mem-Collect-eq scaleR-eq-0-iff simple-functionD*(*2*) *subsetI that*)

have *nonneg-x*: $x \geq 0$ **using** *insert f* **by** *blast*

have $*$: $(\sum y{\in}insert\ x\ F.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ xa\ *_R\ y) = (\sum y{\in}F.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ xa\ *_R\ y) + indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ xa\ *_R\ x$ **for** *xa* **by** (*metis* (*no-types*, *lifting*) *add.commute insert.hyps*(*1*) *insert.hyps*(*4*) *sum.insert*)

have $**$: $\{y \in space\ M.\ (\sum x{\in}insert\ x\ F.\ indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ y *_R\ x) \neq 0\} \subseteq \{y \in space\ M.\ (\sum x{\in}F.\ indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ y *_R\ x) \neq 0\} \cup \{y \in space\ M.\ indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ y *_R\ x \neq 0\}$ **unfolding** $*$ **by** *fastforce*

{

case *1*

show *?case*

proof (*cases* $x = 0$)

case *True*

then show *?thesis* **unfolding** $*$ **using** *insert* **by** *simp*

next

case *False*

have *norm-argument*: $norm\ ((\sum y{\in}F.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ z *_R\ y) + indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ z *_R\ x) = norm\ (\sum y{\in}F.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ z *_R\ y) + norm\ (indicat\text{-}real\ (f\ -`\ \{x\} \cap space\ M)\ z *_R\ x)$ **if** *z*: $z \in space\ M$ **for** *z*

proof (*cases* $f\ z = x$)

case *True*

have $indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ z *_R\ y = 0$ **if** $y \in F$ **for** *y* **using** *True insert z that 1* **unfolding** *indicator-def* **by** *force*

hence $(\sum y{\in}F.\ indicat\text{-}real\ (f\ -`\ \{y\} \cap space\ M)\ z *_R\ y) = 0$ **by** (*meson sum.neutral*)

thus *?thesis* **by** *force*

qed (*force*)

show *?thesis* **using** *False fin-0 fin-1 f norm-argument* **by** (*subst* $*$, *subst add*, *presburger add*: *insert*, *intro insert*, *intro insert*, *fastforce simp add*: *indicator-def intro*!: *insert*(*2*) *f*(*3*), *auto intro*!: *indicator insert nonneg-x*)

qed

next

case *2*

show *?case*

proof (*cases* $x = 0$)

case *True*

then show *?thesis* **unfolding** $*$ **using** *insert* **by** *simp*

next

case *False*

then show *?thesis* **unfolding** $*$ **using** *insert simple-functionD*(*2*)[*OF f*(*1*)] **by** *fast*

qed

next

case *3*

show *?case*

proof (*cases* $x = 0$)

case *True*

then show *?thesis* **unfolding** $*$ **using** *insert* **by** *simp*

next
case *False*
have *emeasure M* $\{y \in$ *space M*. $(\sum x \in$*insert x F*. *indicat-real* $(f -` \{x\}$ $\cap$ *space M*) $y *_R x) \neq 0\} \leq$ *emeasure M* $(\{y \in$ *space M*. $(\sum x \in F$. *indicat-real* $(f -` \{x\} \cap$ *space M*) $y *_R x) \neq 0\} \cup \{y \in$ *space M*. *indicat-real* $(f -` \{x\} \cap$ *space M*) $y *_R x \neq 0\})$
using $**$ *simple-functionD*(2)[*OF insert*(6)] *simple-functionD*(2)[*OF f*(1)] *insert.IH*(2) **by** (*intro emeasure-mono*, *blast*, *simp*)
also have ... $\leq$ *emeasure M* $\{y \in$ *space M*. $(\sum x \in F$. *indicat-real* $(f -` \{x\}$ $\cap$ *space M*) $y *_R x) \neq 0\}$ + *emeasure M* $\{y \in$ *space M*. *indicat-real* $(f -` \{x\} \cap$ *space M*) $y *_R x \neq 0\}$
using *simple-functionD*(2)[*OF insert*(6)] *simple-functionD*(2)[*OF f*(1)] **by** (*intro emeasure-subadditive*, *force*+)
also have ... $< \infty$ **using** *insert*(7) *fin-1*[*OF False*] **by** (*simp add*: *less-top*)
finally show *?thesis* **by** *simp*
qed
next
case (*4* ξ)
thus *?case* **using** *insert nonneg-x f*(3) **by** (*auto simp add*: *scaleR-nonneg-nonneg intro*: *sum-nonneg*)
}
qed
moreover have *simple-function M* (λx. $\sum y \in f$ ` *space M*. *indicat-real* $(f -` \{y\}$ $\cap$ *space M*) $x *_R y$) **using** *calculation* **by** *blast*
moreover have *P* (λx. $\sum y \in f$ ` *space M*. *indicat-real* $(f -` \{y\} \cap$ *space M*) x $*_R y$) **using** *calculation* **by** *blast*
moreover have $\bigwedge x.\ x \in$ *space M* $\Longrightarrow 0 \leq f\ x$ **using** *f*(3) **by** *simp*
ultimately show *?thesis* **by** (*intro cong*[*OF - - - f*(*1*,*2*)], *blast*, *blast*, *fast*) *presburger*+
qed

proposition *integrable-induct'*[*consumes 1*, *case-names base add lim*, *induct pred*: *integrable*]:
fixes *f* :: *'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}
assumes *integrable M f*
assumes *base*: $\bigwedge A\ c.\ A \in$ *sets M* $\Longrightarrow$ *emeasure M A* $< \infty \Longrightarrow P$ (λx. *indicator* $A\ x *_R c$)
assumes *add*: $\bigwedge f\ g$. *integrable M f* $\Longrightarrow P\ f \Longrightarrow$ *integrable M g* $\Longrightarrow P\ g \Longrightarrow P$ ($\lambda x.\ f\ x + g\ x$)
assumes *lim*: $\bigwedge f\ s$. *integrable M f*
$\Longrightarrow$ ($\bigwedge i$. *integrable M* (*s i*))
$\Longrightarrow$ ($\bigwedge i$. *simple-function M* (*s i*))
$\Longrightarrow$ ($\bigwedge i$. *emeasure M* $\{y \in$*space M*. $s\ i\ y \neq 0\} \neq \infty$)
$\Longrightarrow$ ($\bigwedge x.\ x \in$ *space M* $\Longrightarrow (\lambda i.\ s\ i\ x) \longrightarrow f\ x$)
$\Longrightarrow$ ($\bigwedge i\ x.\ x \in$ *space M* $\Longrightarrow$ *norm* $(s\ i\ x) \leq 2 *$ *norm* $(f\ x)$)
$\Longrightarrow$ ($\bigwedge i.\ P\ (s\ i)) \Longrightarrow P\ f$
shows *P f*
proof $-$
have *f*: $f \in$ *borel-measurable M* $(\int^+ x.$ *norm* $(f\ x)\ \partial M) < \infty$ **using** *assms*(*1*)

unfolding *integrable-iff-bounded* **by** *auto*
obtain s **where** s: $\bigwedge i.$ *simple-function* M $(s\ i)$ $\bigwedge x.\ x \in$ *space* $M \Longrightarrow (\lambda i.\ s\ i\ x) \longlonglongrightarrow f\ x$ $\bigwedge i\ x.\ x \in$ *space* $M \Longrightarrow$ *norm* $(s\ i\ x) \leq 2 *$ *norm* $(f\ x)$ **using** *borel-measurable-implies-sequence-metric*[*OF* $f(1)$] **unfolding** *norm-conv-dist* **by** *metis*

{
fix f A
have [*simp*]: P $(\lambda x.\ 0)$ **using** *base*[*of* {} *undefined*] **by** *simp*
have $(\bigwedge i::'b.\ i \in A \Longrightarrow$ *integrable* M $(f\ i::'a \Rightarrow 'b)) \Longrightarrow (\bigwedge i.\ i \in A \Longrightarrow P\ (f\ i)) \Longrightarrow P\ (\lambda x.\ \sum i{\in}A.\ f\ i\ x)$ **by** (*induct A rule*: *infinite-finite-induct*) (*auto intro*!: *add*)
}
note *sum* = *this*

define s' **where** [*abs-def*]: $s'\ i\ z =$ *indicator* (*space* M) $z *_R s\ i\ z$ **for** $i\ z$
hence s'*-eq-s*: $\bigwedge i\ x.\ x \in$ *space* $M \Longrightarrow s'\ i\ x = s\ i\ x$ **by** *simp*

have *sf*[*measurable*]: $\bigwedge i.$ *simple-function* M $(s'\ i)$ **unfolding** s'*-def* **using** $s(1)$ **by** (*intro simple-function-compose2*[**where** $h{=}(*_R)$] *simple-function-indicator*) *auto*

{
fix i
have $\bigwedge z.\ \{y.\ s'\ i\ z = y \wedge y \in s'\ i$ ‘ *space* $M \wedge y \neq 0 \wedge z \in$ *space* $M\}$ = (*if* $z \in$ *space* $M \wedge s'\ i\ z \neq 0$ *then* $\{s'\ i\ z\}$ *else* {}) **by** (*auto simp add*: s'*-def split*: *split-indicator*)
then have $\bigwedge z.\ s'\ i = (\lambda z.\ \sum y{\in}s'\ i$‘*space* $M - \{0\}.$ *indicator* $\{x{\in}$*space* $M.\ s'\ i\ x = y\}\ z *_R y)$ **using** *sf* **by** (*auto simp*: *fun-eq-iff simple-function-def* s'*-def*)
}
note s'*-eq* = *this*

show $P\ f$
proof (*rule lim*)
fix i
have $(\int^+ x.$ *norm* $(s'\ i\ x)\ \partial M) \leq (\int^+ x.$ *ennreal* $(2 *$ *norm* $(f\ x))\ \partial M)$ **using** s **by** (*intro nn-integral-mono*) (*auto simp*: s'*-eq-s*)
also have $\ldots < \infty$ **using** f **by** (*simp add*: *nn-integral-cmult ennreal-mult-less-top ennreal-mult*)
finally have *sbi*: *Bochner-Integration.simple-bochner-integrable* M $(s'\ i)$ **using** *sf* **by** (*intro simple-bochner-integrableI-bounded*) *auto*
thus *integrable* M $(s'\ i)$ *simple-function* M $(s'\ i)$ *emeasure* M $\{y{\in}$*space* $M.\ s'\ i\ y \neq 0\} \neq \infty$ **by** (*auto intro*: *integrableI-simple-bochner-integrable simple-bochner-integrable.cases*)

{
fix x **assume** $x \in$ *space* M $s'\ i\ x \neq 0$
then have *emeasure* M $\{y \in$ *space* $M.\ s'\ i\ y = s'\ i\ x\} \leq$ *emeasure* M $\{y \in$ *space* $M.\ s'\ i\ y \neq 0\}$ **by** (*intro emeasure-mono*) *auto*
also have $\ldots < \infty$ **using** *sbi* **by** (*auto elim*: *simple-bochner-integrable.cases*

simp: *less-top*)
    finally have *emeasure M* $\{y \in$ *space M*. s' *i y* $=$ s' *i x*$\} \neq \infty$ **by** *simp*
  }
  then show *P* (s' *i*) **by** (*subst* s'*-eq*) (*auto intro*!: *sum base simp*: *less-top*)

  fix *x* **assume** $x \in$ *space M*
  thus (λi. s' *i x*) $\longrightarrow$ *f x* **using** *s* **by** (*simp add*: s'*-eq-s*)
  show *norm* (s' *i x*) $\leq$ *2* $*$ *norm* (*f x*) **using** ‹$x \in$ *space M*› *s* **by** (*simp add*: s'*-eq-s*)
 qed *fact*
qed

lemma *finite-nn-integral-imp-ae-finite*:
 fixes *f* :: $'a \Rightarrow$ *ennreal*
 assumes $f \in$ *borel-measurable M* ($\int^+ x.\ f\ x\ \partial M$) $< \infty$
 shows *AE x in M*. *f x* $< \infty$
proof (*rule ccontr*, *goal-cases*)
 case *1*
 let *?A* $=$ *space M* $\cap$ $\{x.\ f\ x = \infty\}$
 have $*$: *emeasure M ?A* $>$ *0* **using** *1 assms*(*1*) **by** (*metis* (*mono-tags*, *lifting*) *assms*(*2*) *eventually-mono infinity-ennreal-def nn-integral-noteq-infinite top.not-eq-extremum*)
 have ($\int^+ x.\ f\ x\ *$ *indicator ?A x* ∂M) $=$ ($\int^+ x.\ \infty\ *$ *indicator ?A x* ∂M) **by** (*metis* (*mono-tags*, *lifting*) *indicator-inter-arith indicator-simps*(*2*) *mem-Collect-eq mult-eq-0-iff*)
 also have ... $= \infty\ *$ *emeasure M ?A* **using** *assms*(*1*) **by** (*intro nn-integral-cmult-indicator*, *simp*)
 also have ... $= \infty$ **using** $*$ **by** *fastforce*
 finally have ($\int^+ x.\ f\ x\ *$ *indicator ?A x* ∂M) $= \infty$ **.**
 moreover have ($\int^+ x.\ f\ x\ *$ *indicator ?A x* ∂M) $\leq$ ($\int^+ x.\ f\ x\ \partial M$) **by** (*intro nn-integral-mono*, *simp add*: *indicator-def*)
 ultimately show *?case* **using** *assms*(*2*) **by** *simp*
qed

lemma *cauchy-L1-AE-cauchy-subseq*:
 fixes *s* :: *nat* $\Rightarrow$ $'a \Rightarrow$ $'b$::{*banach*, *second-countable-topology*}
 assumes [*measurable*]: $\bigwedge n$. *integrable M* (*s n*)
   and $\bigwedge e$. $e > 0 \Longrightarrow \exists N.\ \forall i \geq N.\ \forall j \geq N.$ *LINT x*|*M*. *dist* (*s i x*) (*s j x*) $< e$
 obtains *r* **where** *strict-mono r AE x in M*. *Cauchy* (λi. *s* (*r i*) *x*)
proof−
 have $\exists r.\ \forall n.$ ($\forall i \geq r\ n.\ \forall j \geq r\ n.$ *LINT x*|*M*. *dist* (*s i x*) (*s j x*) $<$ (*1* / *2*) ^ *n*) $\wedge$ (*r* (*Suc n*) $>$ *r n*)
 proof (*intro dependent-nat-choice*, *goal-cases*)
  case *1*
  then show *?case* **using** *assms*(*2*) **by** *presburger*
 next
  case (*2 x n*)
  obtain *N* **where** $*$: *LINT x*|*M*. *dist* (*s i x*) (*s j x*) $<$ (*1* / *2*) ^ *Suc n* **if** $i \geq N$ $j \geq N$ **for** *i j* **using** *assms*(*2*)[*of* (*1* / *2*) ^ *Suc n*] **by** *auto*
  {

fix $i\ j$ **assume** $i \geq max\ N\ (Suc\ x)\ j \geq max\ N\ (Suc\ x)$
hence $integral^L\ M\ (\lambda x.\ dist\ (s\ i\ x)\ (s\ j\ x)) < (1\ /\ 2)\ \widehat{}\ Suc\ n$ **using** $*$ **by** *fastforce*
}
then show *?case* **by** *fastforce*
qed
then obtain r **where** *strict-mono*: *strict-mono* r **and** $\forall i \geq r\ n.\ \forall j \geq\ r\ n.\ LINT\ x|M.\ dist\ (s\ i\ x)\ (s\ j\ x) < (1\ /\ 2)\ \widehat{}\ n$ **for** n **using** *strict-mono-Suc-iff* **by** *blast*
hence *r-is*: $LINT\ x|M.\ dist\ (s\ (r\ (Suc\ n))\ x)\ (s\ (r\ n)\ x) < (1\ /\ 2)\ \widehat{}\ n$ **for** n **by** (*simp add*: *strict-mono-leD*)

define g **where** $g = (\lambda n\ x.\ (\sum i \leq n.\ ennreal\ (dist\ (s\ (r\ (Suc\ i))\ x)\ (s\ (r\ i)\ x))))$
define g' **where** $g' = (\lambda x.\ \sum i.\ ennreal\ (dist\ (s\ (r\ (Suc\ i))\ x)\ (s\ (r\ i)\ x)))$

have *integrable-g*: $(\int^+ x.\ g\ n\ x\ \partial M) < 2$ **for** n
proof $-$
have $(\int^+ x.\ g\ n\ x\ \partial M) = (\int^+ x.\ (\sum i \leq n.\ ennreal\ (dist\ (s\ (r\ (Suc\ i))\ x)\ (s\ (r\ i)\ x)))\ \partial M)$ **using** *g-def* **by** *simp*
also have ... $= (\sum i \leq n.\ (\int^+ x.\ ennreal\ (dist\ (s\ (r\ (Suc\ i))\ x)\ (s\ (r\ i)\ x))\ \partial M))$ **by** (*intro nn-integral-sum*, *simp*)
also have ... $= (\sum i \leq n.\ LINT\ x|M.\ dist\ (s\ (r\ (Suc\ i))\ x)\ (s\ (r\ i)\ x))$ **unfolding** *dist-norm* **using** *assms*(*1*) **by** (*subst nn-integral-eq-integral*[*OF integrable-norm*], *auto*)
also have ... $<$ *ennreal* $(\sum i \leq n.\ (1\ /\ 2)\ \widehat{}\ i)$ **by** (*intro ennreal-lessI*[*OF sum-pos sum-strict-mono*[*OF finite-atMost - r-is*]], *auto*)
also have ... $\leq$ *ennreal 2* **unfolding** *sum-gp0*[*of 1 / 2 n*] **by** (*intro ennreal-leI*, *auto*)
finally show $(\int^+ x.\ g\ n\ x\ \partial M) < 2$ **by** *simp*
qed

have *integrable-g'*: $(\int^+ x.\ g'\ x\ \partial M) \leq 2$
proof $-$
have *incseq* $(\lambda n.\ g\ n\ x)$ **for** x **by** (*intro incseq-SucI*, *auto simp add*: *g-def ennreal-leI*)
hence *convergent* $(\lambda n.\ g\ n\ x)$ **for** x **unfolding** *convergent-def* **using** *LIMSEQ-incseq-SUP* **by** *fast*
hence $(\lambda n.\ g\ n\ x) \longrightarrow g'\ x$ **for** x **unfolding** *g-def g'-def* **by** (*intro summable-iff-convergent'*[*THEN iffD2*, *THEN summable-LIMSEQ'*], *blast*)
hence $(\int^+ x.\ g'\ x\ \partial M) = (\int^+ x.\ liminf\ (\lambda n.\ g\ n\ x)\ \partial M)$ **by** (*metis lim-imp-Liminf trivial-limit-sequentially*)
also have ... $\leq liminf\ (\lambda n.\ \int^+ x.\ g\ n\ x\ \partial M)$ **by** (*intro nn-integral-liminf*, *simp add*: *g-def*)
also have ... $\leq liminf\ (\lambda n.\ 2)$ **using** *integrable-g* **by** (*intro Liminf-mono*) (*simp add*: *order-le-less*)
also have ... $= 2$ **using** *sequentially-bot tendsto-iff-Liminf-eq-Limsup* **by** *blast*
finally show *?thesis* **.**
qed
hence $AE\ x\ in\ M.\ g'\ x < \infty$ **by** (*intro finite-nn-integral-imp-ae-finite*) (*auto simp add*: *order-le-less-trans g'-def*)

moreover have *summable* ($\lambda i.$ *dist* (s (r (*Suc* i)) x) (s (r i) x)) **if** g' $x \neq \infty$ **for** x **using** *that* **unfolding** g'*-def* **by** (*intro summable-suminf-not-top*, *intro zero-le-dist*, *fastforce*)
ultimately have *ae-summable*: *AE x in M*. *summable* ($\lambda i.$ s (r (*Suc* i)) $x - s$ (r i) x) **using** *summable-norm-cancel* **unfolding** *dist-norm* **by** *force*

{
fix x **assume** *summable* ($\lambda i.$ s (r (*Suc* i)) $x - s$ (r i) x)
hence ($\lambda n. \sum i<n.$ s (r (*Suc* i)) $x - s$ (r i) x) $\longrightarrow$ ($\sum i.$ s (r (*Suc* i)) $x - s$ (r i) x) **using** *summable-LIMSEQ* **by** *blast*
moreover have ($\lambda n.$ ($\sum i<n.$ s (r (*Suc* i)) $x - s$ (r i) x)) = ($\lambda n.$ s (r n) $x - s$ (r 0) x) **using** *sum-lessThan-telescope* **by** *fastforce*
ultimately have ($\lambda n.$ s (r n) $x - s$ (r 0) x) $\longrightarrow$ ($\sum i.$ s (r (*Suc* i)) $x - s$ (r i) x) **by** *argo*
hence ($\lambda n.$ s (r n) $x - s$ (r 0) $x + s$ (r 0) x) $\longrightarrow$ ($\sum i.$ s (r (*Suc* i)) $x - s$ (r i) x) $+ s$ (r 0) x **by** (*intro isCont-tendsto-compose*[*of* - $\lambda z.$ $z + s$ (r 0) x], *auto*)
hence *Cauchy* ($\lambda n.$ s (r n) x) **by** (*simp add*: *LIMSEQ-imp-Cauchy*)
}

hence *AE x in M*. *Cauchy* ($\lambda i.$ s (r i) x) **using** *ae-summable* **by** *fast*
thus *?thesis* **by** (*rule that*[*OF strict-mono*(*1*)])
qed

lemma *integrable-max*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*}
assumes *fg*[*measurable*]: *integrable M f integrable M g*
shows *integrable M* ($\lambda x.$ *max* (f x) (g x))
proof (*rule Bochner-Integration.integrable-bound*)
{
fix x y :: $'b$
have *norm* (*max* x y) $\leq$ *max* (*norm* x) (*norm* y) **by** *linarith*
also have ... $\leq$ *norm* x + *norm* y **by** *simp*
finally have *norm* (*max* x y) $\leq$ *norm* (*norm* x + *norm* y) **by** *simp*
}
thus *AE x in M*. *norm* (*max* (f x) (g x)) $\leq$ *norm* (*norm* (f x) + *norm* (g x)) **by** *simp*
qed (*auto intro*: *Bochner-Integration.integrable-add*[*OF integrable-norm*[*OF fg*(*1*)] *integrable-norm*[*OF fg*(*2*)]]])

lemma *integrable-min*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*}
assumes [*measurable*]: *integrable M f integrable M g*
shows *integrable M* ($\lambda x.$ *min* (f x) (g x))
proof −
have *norm* (*min* (f x) (g x)) $\leq$ *norm* (f x) $\vee$ *norm* (*min* (f x) (g x)) $\leq$ *norm* (g x) **for** x **by** *linarith*
thus *?thesis* **by** (*intro integrable-bound*[*OF integrable-max*[*OF integrable-norm*(*1*,*1*), *OF assms*]], *auto*)
qed

lemma *integral-nonneg-AE-banach*:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes [*measurable*]: $f \in$ *borel-measurable M* **and** *nonneg*: *AE x in M*. $0 \leq f\ x$
shows $0 \leq integral^L\ M\ f$
proof *cases*
assume *integrable*: *integrable M f*
hence *max*: ($\lambda x.$ *max 0* ($f\ x$)) $\in$ *borel-measurable M* $\bigwedge x.\ 0 \leq$ *max 0* ($f\ x$) *integrable M* ($\lambda x.$ *max 0* ($f\ x$)) **by** *auto*
hence $0 \leq integral^L\ M$ ($\lambda x.$ *max 0* ($f\ x$))
proof –
obtain s **where** $*$: $\bigwedge i.$ *simple-function M* ($s\ i$)
$\bigwedge i.$ *emeasure M* $\{y \in$ *space M*. $s\ i\ y \neq 0\} \neq \infty$
$\bigwedge x.\ x \in$ *space M* $\Longrightarrow$ ($\lambda i.\ s\ i\ x$) $\longrightarrow f\ x$
$\bigwedge x\ i.\ x \in$ *space M* $\Longrightarrow$ *norm* ($s\ i\ x$) $\leq 2 *$ *norm* ($f\ x$) **using** *integrable-implies-simple-function-sequence*[*OF integrable*] **by** *blast*
have *simple*: $\bigwedge i.$ *simple-function M* ($\lambda x.$ *max 0* ($s\ i\ x$)) **using** $*$ **by** *fast*
have $\bigwedge i.\ \{y \in$ *space M*. *max 0* ($s\ i\ y$) $\neq 0\} \subseteq \{y \in$ *space M*. $s\ i\ y \neq 0\}$ **unfolding** *max-def* **by** *force*
moreover have $\bigwedge i.\ \{y \in$ *space M*. $s\ i\ y \neq 0\} \in$ *sets M* **using** $*$ **by** *measurable*
ultimately have $\bigwedge i.$ *emeasure M* $\{y \in$ *space M*. *max 0* ($s\ i\ y$) $\neq 0\} \leq$ *emeasure M* $\{y \in$ *space M*. $s\ i\ y \neq 0\}$ **by** (*rule emeasure-mono*)
hence $**$:$\bigwedge i.$ *emeasure M* $\{y \in$ *space M*. *max 0* ($s\ i\ y$) $\neq 0\} \neq \infty$ **using** $*(2)$ **by** (*auto intro*: *order.strict-trans1 simp add*: *top.not-eq-extremum*)
have $\bigwedge x.\ x \in$ *space M* $\Longrightarrow$ ($\lambda i.$ *max 0* ($s\ i\ x$)) $\longrightarrow$ *max 0* ($f\ x$) **using** $*(3)$ *tendsto-max* **by** *blast*
moreover have $\bigwedge x\ i.\ x \in$ *space M* $\Longrightarrow$ *norm* (*max 0* ($s\ i\ x$)) $\leq$ *norm* ($2 *_R f\ x$) **using** $*(4)$ **unfolding** *max-def* **by** *auto*
ultimately have *tendsto*: ($\lambda i.\ integral^L\ M$ ($\lambda x.$ *max 0* ($s\ i\ x$))) $\longrightarrow integral^L\ M$ ($\lambda x.$ *max 0* ($f\ x$))
using *borel-measurable-simple-function simple integrable* **by** (*intro integral-dominated-convergence*[*OF max*(*1*) - *integrable-norm*[*OF integrable-scaleR-right*], *of - 2 f*], *blast*+)
{
fix h :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assume *simple-function M h emeasure M* $\{y \in$ *space M*. $h\ y \neq 0\} \neq \infty$ $\bigwedge x.\ x \in$ *space M* $\longrightarrow h\ x \geq 0$
hence $*$: $integral^L\ M\ h \geq 0$
proof (*induct rule*: *simple-integrable-function-induct-nonneg*)
case (*cong f g*)
then show *?case* **using** *Bochner-Integration.integral-cong* **by** *force*
next
case (*indicator A y*)
hence $A \neq \{\} \Longrightarrow y \geq 0$ **using** *sets.sets-into-space* **by** *fastforce*
then show *?case* **using** *indicator* **by** (*cases* $A = \{\}$, *auto simp add*:

scaleR-nonneg-nonneg)
      next
        case (*add f g*)
        then show *?case* **by** (*simp add*: *integrable-simple-function*)
      qed
    }
    thus *?thesis* **using** *LIMSEQ-le-const*[*OF tendsto*, *of 0*] ** *simple* **by** *fastforce*
  qed
  also have ... = $integral^L$ *M f* **using** *nonneg* **by** (*auto intro*: *integral-cong-AE*)
  finally show *?thesis* **.**
qed (*simp add*: *not-integrable-integral-eq*)

lemma *integral-mono-AE-banach*:
  fixes *f g* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *integrable M f integrable M g AE x in M*. *f x* ≤ *g x*
  shows $integral^L$ *M f* ≤ $integral^L$ *M g*
  using *integral-nonneg-AE-banach*[*of* λ*x*. *g x* − *f x*] *assms Bochner-Integration.integral-diff*[*OF assms*(*1*,*2*)] **by** *force*

lemma *integral-mono-banach*:
  fixes *f g* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *integrable M f integrable M g* ⋀*x*. *x* ∈ *space M* ⟹ *f x* ≤ *g x*
  shows $integral^L$ *M f* ≤ $integral^L$ *M g*
  using *integral-mono-AE-banach assms* **by** *blast*

end
theory *Set-Integral-Addendum*
  imports *HOL−Analysis.Set-Integral Bochner-Integration-Addendum*
  begin

lemma *set-integral-scaleR-left*:
  assumes *A* ∈ *sets M c* ≠ *0* ⟹ *integrable M f*
  shows *LINT t*:*A*|*M*. *f t* $*_R$ *c* = (*LINT t*:*A*|*M*. *f t*) $*_R$ *c*
  unfolding *set-lebesgue-integral-def*
  using *integrable-mult-indicator*[*OF assms*]
  by (*subst integral-scaleR-left*[*symmetric*], *auto*)

lemma *nn-set-integral-eq-set-integral*:
  assumes [*measurable*]:*integrable M f*
      and *AE x* ∈ *A in M*. *0* ≤ *f x A* ∈ *sets M*
    shows ($\int^+$*x*∈*A*. *f x* ∂*M*) = ($\int$ *x* ∈ *A*. *f x* ∂*M*)
proof−
  have ($\int^+$*x*. *indicator A x* $*_R$ *f x* ∂*M*) = ($\int$ *x* ∈ *A*. *f x* ∂*M*)
  unfolding *set-lebesgue-integral-def* **using** *assms*(*2*) **by** (*intro nn-integral-eq-integral*[*of* - λ*x*. *indicat-real A x* $*_R$ *f x*], *blast intro*: *assms integrable-mult-indicator*, *fastforce*)
  moreover have ($\int^+$*x*. *indicator A x* $*_R$ *f x* ∂*M*) = ($\int^+$*x*∈*A*. *f x* ∂*M*) **by** (*metis ennreal-0 indicator-simps*(*1*) *indicator-simps*(*2*) *mult.commute mult-1 mult-zero-left*

real-scaleR-def)
  ultimately show *?thesis* **by** *argo*
qed

lemma *set-integral-restrict-space*:
  fixes f :: *'a* ⇒ *'b*::{*banach*, *second-countable-topology*}
  assumes $\Omega \cap$ *space M* ∈ *sets M*
  shows *set-lebesgue-integral* (*restrict-space M* Ω) *A f* = *set-lebesgue-integral M A* (λ*x*. *indicator* Ω *x* $*_R$ *f x*)
  unfolding *set-lebesgue-integral-def*
  by (*subst integral-restrict-space*, *auto intro*!: *integrable-mult-indicator assms simp*: *mult.commute*)

lemma *set-integral-const*:
  fixes *c* :: *'b*::{*banach*, *second-countable-topology*}
  assumes *A* ∈ *sets M emeasure M A* ≠ ∞
  shows *set-lebesgue-integral M A* (λ-. *c*) = *measure M A* $*_R$ *c*
  unfolding *set-lebesgue-integral-def*
  using *assms* **by** (*metis has-bochner-integral-indicator has-bochner-integral-integral-eq infinity-ennreal-def less-top*)

lemma *set-integral-mono-banach*:
  fixes *f g* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *set-integrable M A f set-integrable M A g*
    ⋀*x*. *x* ∈ *A* ⟹ *f x* ≤ *g x*
  shows (*LINT x*:*A*|*M*. *f x*) ≤ (*LINT x*:*A*|*M*. *g x*)
  using *assms* **unfolding** *set-integrable-def set-lebesgue-integral-def*
  by (*auto intro*: *integral-mono-banach split*: *split-indicator*)

lemma *set-integral-mono-AE-banach*:
  fixes *f g* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *set-integrable M A f set-integrable M A g AE x*∈*A in M*. *f x* ≤ *g x*
  shows *set-lebesgue-integral M A f* ≤ *set-lebesgue-integral M A g* **using** *assms* **unfolding** *set-lebesgue-integral-def* **by** (*auto simp add*: *set-integrable-def intro*!: *integral-mono-AE-banach*[*of M* λ*x*. *indicator A x* $*_R$ *f x* λ*x*. *indicator A x* $*_R$ *g x*], *simp add*: *indicator-def*)

end
theory *Sigma-Finite-Measure-Addendum*
imports *Set-Integral-Addendum*
begin

lemma *balls-countable-basis*:
  obtains *D* :: *'a* :: {*metric-space*, *second-countable-topology*} *set*
  where *topological-basis* (*case-prod ball* ‘ (*D* × (ℚ ∩ {*0*<..})))
    and *countable D*

and $D \neq \{\}$
proof $-$
obtain D :: *'a set* **where** *dense-subset*: *countable D* $D \neq \{\}$ $[\![$*open U*; $U \neq \{\}]\!]$
$\Longrightarrow \exists\, y \in D.\ y \in U$ **for** *U* **using** *countable-dense-exists* **by** *blast*
have *topological-basis* (*case-prod ball* ‘ ($D \times (\mathbb{Q} \cap \{0<..\})$))
proof (*intro topological-basis-iff*[*THEN iffD2*], *fast*, *clarify*)
fix *U* **and** *x* :: *'a* **assume** *asm*: *open U* $x \in U$
obtain *e* **where** *e*: $e > 0$ *ball x e* $\subseteq U$ **using** *asm openE* **by** *blast*
obtain *y* **where** *y*: $y \in D$ $y \in$ *ball x* (e / 3) **using** *dense-subset*(*3*)[*OF open-ball*,
of x e / *3*] *centre-in-ball*[*THEN iffD2*, *OF divide-pos-pos*[*OF e*(*1*), *of 3*]] **by** *force*
obtain *r* **where** *r*: $r \in \mathbb{Q} \cap \{e/3<..<e/2\}$ **unfolding** *Rats-def* **using** *of-rat-dense*[*OF*
divide-strict-left-mono[*OF - e*(*1*)], *of 2 3*] **by** *auto*

have $*$: $x \in$ *ball y r* **using** *r y* **by** (*simp add*: *dist-commute*)
hence *ball y r* $\subseteq U$ **using** *r* **by** (*intro order-trans*[*OF - e*(*2*)], *simp*, *metric*)
moreover have *ball y r* $\in$ (*case-prod ball* ‘ ($D \times (\mathbb{Q} \cap \{0<..\})$)) **using** *y*(*1*)
r **by** *force*
ultimately show $\exists\, B' \in$(*case-prod ball* ‘ ($D \times (\mathbb{Q} \cap \{0<..\})$)). $x \in B' \wedge B' \subseteq$
U **using** $*$ **by** *meson*
qed
thus *?thesis* **using** *that dense-subset* **by** *blast*
qed

context *sigma-finite-measure*
begin

lemma *sigma-finite-measure-induct*[*case-names finite-measure*, *consumes 0*]:
assumes $\bigwedge$(*N* :: *'a measure*) Ω. *finite-measure N*
$\Longrightarrow$ N = *restrict-space M* Ω
$\Longrightarrow$ $\Omega \in$ *sets M*
$\Longrightarrow$ *emeasure N* $\Omega \neq \infty$
$\Longrightarrow$ *emeasure N* $\Omega \neq 0$
$\Longrightarrow$ *almost-everywhere N Q*
and [*measurable*]: *Measurable.pred M Q*
shows *almost-everywhere M Q*
proof $-$
have $*$: *almost-everywhere N Q* **if** *finite-measure N* N = *restrict-space M* Ω Ω
$\in$ *sets M emeasure N* $\Omega \neq \infty$ **for** *N* Ω **using** *that* **by** (*cases emeasure N* $\Omega = 0$,
auto intro: *emeasure-0-AE assms*(*1*))

obtain *A* :: *nat* $\Rightarrow$ *'a set* **where** *A*: *range A* $\subseteq$ *sets M* ($\bigcup i.\ A\ i$) = *space M* **and**
emeasure-finite: *emeasure M* (*A i*) $\neq \infty$ **for** *i* **using** *sigma-finite* **by** *metis*
note *A*(*1*)[*measurable*]
have *space-restr*: *space* (*restrict-space M* (*A i*)) = *A i* **for** *i* **unfolding** *space-restrict-space*
by *simp*
{
fix *i*
have $*$: $\{x \in A\ i \cap$ *space M*. $Q\ x\}$ = $\{x \in$ *space M*. $Q\ x\} \cap (A\ i)$ **by** *fast*
have *Measurable.pred* (*restrict-space M* (*A i*)) *Q* **using** *A* **by** (*intro measurableI*,

auto simp add: *space-restr intro*!: *sets-restrict-space-iff* [*THEN iffD2*], *measurable*, *auto*)
}
note *this*[*measurable*]
{
fix *i*
have *finite-measure* (*restrict-space M* (*A i*)) **using** *emeasure-finite* **by** (*intro finite-measureI*, *subst space-restr*, *subst emeasure-restrict-space*, *auto*)
hence *emeasure* (*restrict-space M* (*A i*)) {*x* ∈ *A i*. ¬*Q x*} = *0* **using** *emeasure-finite* **by** (*intro AE-iff-measurable*[*THEN iffD1*, *OF - - ∗*], *measurable*, *subst space-restr*[*symmetric*], *intro sets.top*, *auto simp add*: *emeasure-restrict-space*)
hence *emeasure M* {*x* ∈ *A i*. ¬ *Q x*} = *0* **by** (*subst emeasure-restrict-space*[*symmetric*], *auto*)
}
hence *emeasure M* (⋃*i*. {*x* ∈ *A i*. ¬ *Q x*}) = *0* **by** (*intro emeasure-UN-eq-0*, *auto*)
moreover have (⋃*i*. {*x* ∈ *A i*. ¬ *Q x*}) = {*x* ∈ *space M*. ¬ *Q x*} **using** *A* **by** *auto*
ultimately show *?thesis* **by** (*intro AE-iff-measurable*[*THEN iffD2*], *auto*)
qed

lemma *averaging-theorem*:
fixes *f*::- ⇒ ′*b*::{*second-countable-topology*, *banach*}
assumes [*measurable*]:*integrable M f*
and *closed*: *closed S*
and ⋀*A*. *A* ∈ *sets M* ⟹ *measure M A* > *0* ⟹ (*1* / *measure M A*) $*_R$ *set-lebesgue-integral M A f* ∈ *S*
shows *AE x in M*. *f x* ∈ *S*
proof (*induct rule*: *sigma-finite-measure-induct*)
case (*finite-measure N* Ω)

interpret *finite-measure N* **by** (*rule finite-measure*)

have *integrable*[*measurable*]: *integrable N f* **using** *assms finite-measure* **by** (*auto simp*: *integrable-restrict-space integrable-mult-indicator*)
have *average*: (*1* / *Sigma-Algebra.measure N A*) $*_R$ *set-lebesgue-integral N A f* ∈ *S* **if** *A* ∈ *sets N measure N A* > *0* **for** *A*
proof −
have ∗: *A* ∈ *sets M* **using** *that* **by** (*simp add*: *sets-restrict-space-iff finite-measure*)
have *A* = *A* ∩ Ω **by** (*metis finite-measure*(*2*,*3*) *inf.orderE sets.sets-into-space space-restrict-space that*(*1*))
hence *set-lebesgue-integral N A f* = *set-lebesgue-integral M A f* **unfolding** *finite-measure* **by** (*subst set-integral-restrict-space*, *auto simp add*: *finite-measure set-lebesgue-integral-def indicator-inter-arith*[*symmetric*])
moreover have *measure N A* = *measure M A* **using** *that* **by** (*auto intro*!: *measure-restrict-space simp add*: *finite-measure sets-restrict-space-iff*)
ultimately show *?thesis* **using** *that* ∗ *assms*(*3*) **by** *presburger*
qed

obtain *D* :: *'b set* **where** *balls-basis*: *topological-basis* (*case-prod ball* ‘ ($D \times$ ($\mathbb{Q} \cap \{0<..\}$))) **and** *countable-D*: *countable D* **using** *balls-countable-basis* **by** *blast*
have *countable-balls*: *countable* (*case-prod ball* ‘ ($D \times$ ($\mathbb{Q} \cap \{0<..\}$))) **using** *countable-rat countable-D* **by** *blast*

obtain *B* **where** *B-balls*: $B \subseteq$ *case-prod ball* ‘ ($D \times$ ($\mathbb{Q} \cap \{0<..\}$)) $\bigcup B = -S$ **using** *topological-basis*[*THEN iffD1*, *OF balls-basis*] *open-Compl*[*OF assms*(*2*)] **by** *meson*
hence *countable-B*: *countable B* **using** *countable-balls countable-subset* **by** *fast*

define *b* **where** $b =$ *from-nat-into* ($B \cup \{\{\}\}$)
have $B \cup \{\{\}\} \neq \{\}$ **by** *simp*
have *range-b*: *range b* $= B \cup \{\{\}\}$ **using** *countable-B* **by** (*auto simp add*: *b-def intro*!: *range-from-nat-into*)
have *open-b*: *open* (*b i*) **for** *i* **unfolding** *b-def* **using** *B-balls open-ball from-nat-into*[*of* $B \cup \{\{\}\}$ *i*] **by** *force*
have *Union-range-b*: $\bigcup$(*range b*) $= -S$ **using** *B-balls range-b* **by** *simp*

{
fix *v r* **assume** *ball-in-Compl*: *ball v r* $\subseteq -S$
define *A* **where** $A = f$ −‘ *ball v r* $\cap$ *space N*
have *dist-less*: *dist* (*f x*) *v* $<$ *r* **if** $x \in A$ **for** *x* **using** *that* **unfolding** *A-def vimage-def* **by** (*simp add*: *dist-commute*)
hence *AE-less*: *AE* $x \in A$ *in N*. *norm* ($f\ x - v$) $<$ *r* **by** (*auto simp add*: *dist-norm*)
have $*$: $A \in$ *sets N* **unfolding** *A-def* **by** *simp*
have *emeasure N A* $= 0$
proof −
{
assume *asm*: *emeasure N A* > 0
hence *measure-pos*: *measure N A* > 0 **unfolding** *emeasure-eq-measure* **by** *simp*
hence (*1* / *measure N A*) $*_R$ *set-lebesgue-integral N A f* $- v =$ (*1* / *measure N A*) $*_R$ *set-lebesgue-integral N A* ($\lambda x.\ f\ x - v$) **using** *integrable integrable-const* $*$ **by** (*subst set-integral-diff*(*2*), *auto simp add*: *set-integrable-def set-integral-const*[*OF* $*$] *algebra-simps intro*!: *integrable-mult-indicator*)
moreover have *norm* ($\int x{\in}A.\ (f\ x - v)\partial N$) $\leq$ ($\int x{\in}A.$ *norm* ($f\ x - v$)∂N) **using** $*$ **by** (*auto intro*!: *integral-norm-bound*[*of N* $\lambda x.$ *indicator A x* $*_R$ ($f\ x - v$), *THEN order-trans*] *integrable-mult-indicator integrable simp add*: *set-lebesgue-integral-def*)
ultimately have *norm* ((*1* / *measure N A*) $*_R$ *set-lebesgue-integral N A f* $- v$) $\leq$ *set-lebesgue-integral N A* ($\lambda x.$ *norm* ($f\ x - v$)) / *measure N A* **using** *asm* **by** (*auto intro*: *divide-right-mono*)
also have ... $<$ *set-lebesgue-integral N A* ($\lambda x.\ r$) / *measure N A*
unfolding *set-lebesgue-integral-def*
using *asm* $*$ *integrable integrable-const AE-less measure-pos*
by (*intro divide-strict-right-mono integral-less-AE*[*of - - A*] *integrable-mult-indicator*)
(*fastforce simp add*: *dist-less dist-norm indicator-def*)+

also have ... = r **using** $*$ *measure-pos* **by** (*simp add*: *set-integral-const*)
finally have *dist* ((*1* / *measure N A*) $*_R$ *set-lebesgue-integral N A f*) $v < r$ **by** (*subst dist-norm*)
hence *False* **using** *average*[*OF* $*$ *measure-pos*] **by** (*metis ComplD dist-commute in-mono mem-ball ball-in-Compl*)
}
thus *?thesis* **by** *fastforce*
qed
}
note $*$ = *this*
{
fix b' **assume** $b' \in B$
hence *ball-subset-Compl*: $b' \subseteq -S$ **and** *ball-radius-pos*: $\exists\, v \in D.\ \exists\, r{>}0.\ b' =$ *ball v r* **using** *B-balls* **by** (*blast*, *fast*)
}
note $**$ = *this*
hence *emeasure N* (f −‘ $b\ i \cap$ *space N*) = *0* **for** i **by** (*cases* $b\ i = \{\}$, *simp*) (*metis UnE singletonD* $*$ *range-b*[*THEN eq-refl*, *THEN range-subsetD*])
hence *emeasure N* ($\bigcup i.\ f$ −‘ $b\ i \cap$ *space N*) = *0* **using** *open-b* **by** (*intro emeasure-UN-eq-0*) *fastforce*+
moreover have ($\bigcup i.\ f$ −‘ $b\ i \cap$ *space N*) = f −‘ ($\bigcup$(*range b*)) $\cap$ *space N* **by** *blast*
ultimately have *emeasure N* (f −‘ ($-S$) $\cap$ *space N*) = *0* **using** *Union-range-b* **by** *argo*
hence *AE x in N*. $f\ x \notin -S$ **using** *open-Compl*[*OF assms*(*2*)] **by** (*intro AE-iff-measurable*[*THEN iffD2*], *auto*)
thus *?case* **by** *force*
qed (*simp add*: *pred-sets2*[*OF borel-closed*] *assms*(*2*))

lemma *density-nonneg*:
fixes f::- ⇒ $'b$::{*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable M f*
and $\bigwedge A.\ A \in$ *sets M* $\Longrightarrow$ *set-lebesgue-integral M A f* $\geq$ *0*
shows *AE x in M*. $f\ x \geq 0$
using *averaging-theorem*[*OF assms*(*1*), *of* {*0*..}, *OF closed-atLeast*] *assms*(*2*)
by (*simp add*: *scaleR-nonneg-nonneg*)

lemma *density-zero*:
fixes f::$'a$ ⇒ $'b$::{*second-countable-topology*, *banach*}
assumes *integrable M f*
and *density-0*: $\bigwedge A.\ A \in$ *sets M* $\Longrightarrow$ *set-lebesgue-integral M A f* = *0*
shows *AE x in M*. $f\ x = 0$
using *averaging-theorem*[*OF assms*(*1*), *of* {*0*}] *assms*(*2*)
by (*simp add*: *scaleR-nonneg-nonneg*)

lemma *density-unique*:
fixes $f\ f'$::$'a$ ⇒ $'b$::{*second-countable-topology*, *banach*}
assumes *integrable M f integrable M f'*
and *density-eq*: $\bigwedge A.\ A \in$ *sets M* $\Longrightarrow$ *set-lebesgue-integral M A f* = *set-lebesgue-integral*

M A f′
  shows *AE x in M. f x = f′ x*
proof−
  {
    fix *A* **assume** *asm*: *A ∈ sets M*
      hence *LINT x|M. indicat-real A x ∗R (f x − f′ x) = 0* **using** *density-eq assms(1,2)* **by** (*simp add*: *set-lebesgue-integral-def algebra-simps Bochner-Integration.integral-diff*[*OF integrable-mult-indicator*(*1*,*1*)])
  }
 thus *?thesis* **using** *density-zero*[*OF Bochner-Integration.integrable-diff*[*OF assms*(*1*,*2*)]] **by** (*simp add*: *set-lebesgue-integral-def*)
qed

lemma *integral-nonneg-AE-eq-0-iff-AE*:
  fixes *f* :: *′a ⇒ ′b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *f*[*measurable*]: *integrable M f* **and** *nonneg*: *AE x in M. 0 ≤ f x*
  shows *integral^L M f = 0 ⟷ (AE x in M. f x = 0)*
proof
  assume ∗: *integral^L M f = 0*
  {
    fix *A* **assume** *asm*: *A ∈ sets M*
    have *0 ≤ integral^L M (λx. indicator A x ∗R f x)* **using** *nonneg* **by** (*subst integral-zero*[*of M*, *symmetric*], *intro integral-mono-AE-banach integrable-mult-indicator asm f integrable-zero*, *auto simp add*: *indicator-def*)
    moreover have ... *≤ integral^L M f* **using** *nonneg* **by** (*intro integral-mono-AE-banach integrable-mult-indicator asm f*, *auto simp add*: *indicator-def*)
    ultimately have *set-lebesgue-integral M A f = 0* **unfolding** *set-lebesgue-integral-def* **using** ∗ **by** *force*
  }
  thus *AE x in M. f x = 0* **by** (*intro density-zero f*, *blast*)
qed (*auto simp add*: *integral-eq-zero-AE*)

lemma *integral-eq-mono-AE-eq-AE*:
  fixes *f g* :: *′a ⇒ ′b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
  assumes *integrable M f integrable M g integral^L M f = integral^L M g AE x in M. f x ≤ g x*
  shows *AE x in M. f x = g x*
proof −
  define *h* **where** *h = (λx. g x − f x)*
  have *AE x in M. h x = 0* **unfolding** *h-def* **using** *assms* **by** (*subst integral-nonneg-AE-eq-0-iff-AE*[*symmetric*]) *auto*
  then show *?thesis* **unfolding** *h-def* **by** *auto*
qed

end

end

theory *Filtration*
imports *HOL−Probability.Conditional-Expectation HOL−Probability.Stopping-Time Measure-Space-Addendum*
begin

1.3 Filtered Sigma Finite Measure

locale *filtered-sigma-finite-measure* = *sigma-finite-measure M* + *filtration space M F* **for** *M* **and** $F :: {'t} :: \{\textit{second-countable-topology}, \textit{linorder-topology}, \textit{order-bot}\} \Rightarrow {'a}\ \textit{measure}$ +
  assumes *subalgebra*: $\bigwedge i.\ \textit{subalgebra}\ M\ (F\ i)$
    and *sigma-finite*: *sigma-finite-measure* (*restr-to-subalg M* (*F bot*))

locale *ennreal-filtered-sigma-finite-measure* = *filtered-sigma-finite-measure M F* **for** *M* **and** $F :: \textit{ennreal} \Rightarrow$ -
locale *nat-filtered-sigma-finite-measure* = *filtered-sigma-finite-measure M F* **for** *M* **and** $F :: \textit{nat} \Rightarrow$ -

sublocale *filtered-sigma-finite-measure* $\subseteq$ *sigma-finite-subalgebra M F i* **by** (*metis bot.extremum sigma-finite sigma-finite-subalgebra.intro subalgebra sets-F-mono sigma-finite-subalgebra.nested-s subalgebra-def*)

1.4 Natural Filtration

definition *natural-filtration* :: $'a\ \textit{measure} \Rightarrow {'s}\ \textit{measure} \Rightarrow ({'t} \Rightarrow {'a} \Rightarrow {'s}) \Rightarrow {'t} :: \{\textit{second-countable-topology}, \textit{linorder-topology}, \textit{order-bot}\} \Rightarrow {'a}\ \textit{measure}$ **where**
  natural-filtration M N Y = (λt. *restr-to-subalg M* (*sigma-gen* (*space M*) *N* $\{Y\ i \mid i.\ i \le t\}$))

lemma
  assumes $\bigwedge i.\ Y\ i \in M \to_M N$
  shows *sets-natural-filtration*[*simp*]: *sets* (*natural-filtration M N Y t*) = *sigma-sets* (*space M*) ($\bigcup i \le t.\ \{Y\ i\ -`\ A \cap \textit{space}\ M \mid A.\ A \in N\}$)
   and *space-natural-filtration*[*simp*]: *space* (*natural-filtration M N Y t*) = *space M*
  by (*standard*; (*subst natural-filtration-def*, *subst sets-restr-to-subalg*)) (*auto simp add*: *natural-filtration-def space-restr-to-subalg subalgebra-def intro*!: *sets.sigma-sets-subset measurable-sets*[*OF assms*] *sigma-sets-mono*)

end
theory *Conditional-Expectation-Banach*
imports *HOL−Probability.Conditional-Expectation Sigma-Finite-Measure-Addendum Bochner-Integration-Addendum Elementary-Metric-Spaces-Addendum*
begin

definition *has-cond-exp* :: $'a\ \textit{measure} \Rightarrow {'a}\ \textit{measure} \Rightarrow ({'a} \Rightarrow {'b}) \Rightarrow ({'a} \Rightarrow {'b}::\{\textit{real-normed-vector}, \textit{second-countable-topology}\}) \Rightarrow \textit{bool}$ **where**
  has-cond-exp M F f g = $((\forall A \in \textit{sets}\ F.\ (\int x \in A.\ f\ x\ \partial M) = (\int x \in A.\ g\ x\ \partial M))$
          $\wedge$ *integrable M f*
          $\wedge$ *integrable M g*

$\wedge$ *g* $\in$ *borel-measurable F*)

lemma *has-cond-expI'*[*intro*]:
assumes $\bigwedge A$. $A \in$ *sets F* $\Longrightarrow$ ($\int$ $x \in A$. *f x* ∂M) = ($\int$ $x \in A$. *g x* ∂M)
integrable M f
integrable M g
g $\in$ *borel-measurable F*
shows *has-cond-exp M F f g*
using *assms* **unfolding** *has-cond-exp-def* **by** *simp*

lemma *has-cond-expD*:
assumes *has-cond-exp M F f g*
shows $\bigwedge A$. $A \in$ *sets F* $\Longrightarrow$ ($\int$ $x \in A$. *f x* ∂M) = ($\int$ $x \in A$. *g x* ∂M)
integrable M f
integrable M g
g $\in$ *borel-measurable F*
using *assms* **unfolding** *has-cond-exp-def* **by** *simp+*

definition *cond-exp* :: *'a measure* $\Rightarrow$ *'a measure* $\Rightarrow$ (*'a* $\Rightarrow$ *'b*) $\Rightarrow$ (*'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}) **where**
cond-exp M F f = (*if* $\exists$ *g*. *has-cond-exp M F f g then* (*SOME g*. *has-cond-exp M F f g*) *else* (λ-. *0*))

lemma *borel-measurable-cond-exp*[*measurable*]: *cond-exp M F f* $\in$ *borel-measurable F*
by (*metis cond-exp-def someI has-cond-exp-def borel-measurable-const*)

lemma *integrable-cond-exp*[*intro*]: *integrable M* (*cond-exp M F f*)
by (*metis cond-exp-def has-cond-expD*(*3*) *integrable-zero someI*)

lemma *set-integrable-cond-exp*[*intro*]:
assumes *A* $\in$ *sets M*
shows *set-integrable M A* (*cond-exp M F f*) **using** *integrable-mult-indicator*[*OF assms integrable-cond-exp*, *of F f*] **by** (*auto simp add*: *set-integrable-def intro*!: *integrable-mult-indicator*[*OF assms integrable-cond-exp*])

context *sigma-finite-subalgebra*
begin

lemma *borel-measurable-cond-exp'*[*measurable*]: *cond-exp M F f* $\in$ *borel-measurable M*
by (*metis cond-exp-def someI has-cond-exp-def borel-measurable-const subalg measurable-from-subalg*)

lemma *cond-exp-null*:
assumes $\nexists$ *g*. *has-cond-exp M F f g*
shows *cond-exp M F f* = (λ-. *0*)
unfolding *cond-exp-def* **using** *assms* **by** *argo*

lemma *has-cond-exp-charact*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*, *banach*}
assumes *has-cond-exp* $M\ F\ f\ g$
shows *has-cond-exp* $M\ F\ f$ (*cond-exp* $M\ F\ f$)
$AE\ x\ in\ M.$ *cond-exp* $M\ F\ f\ x = g\ x$
proof −
show *cond-exp*: *has-cond-exp* $M\ F\ f$ (*cond-exp* $M\ F\ f$) **using** *assms someI cond-exp-def* **by** *metis*
let *?MF* = *restr-to-subalg* $M\ F$
interpret *sigma-finite-measure ?MF* **by** (*rule sigma-fin-subalg*)
{
fix A **assume** $A \in$ *sets ?MF*
then have [*measurable*]: $A \in$ *sets* F **using** *sets-restr-to-subalg*[*OF subalg*] **by** *simp*
have ($\int x \in A.\ g\ x\ \partial$*?MF*) = ($\int x \in A.\ g\ x\ \partial M$) **using** *assms subalg* **by** (*auto simp add*: *integral-subalgebra2 set-lebesgue-integral-def dest*!: *has-cond-expD*)
also have ... = ($\int x \in A.$ *cond-exp* $M\ F\ f\ x\ \partial M$) **using** *assms cond-exp* **by** (*simp add*: *has-cond-exp-def*)
also have ... = ($\int x \in A.$ *cond-exp* $M\ F\ f\ x\ \partial$*?MF*) **using** *subalg* **by** (*auto simp add*: *integral-subalgebra2 set-lebesgue-integral-def*)
finally have ($\int x \in A.\ g\ x\ \partial$*?MF*) = ($\int x \in A.$ *cond-exp* $M\ F\ f\ x\ \partial$*?MF*) **by** *simp*
}
hence $AE\ x\ in$ *?MF*. *cond-exp* $M\ F\ f\ x = g\ x$ **using** *cond-exp assms subalg* **by** (*intro density-unique*, *auto dest*: *has-cond-expD intro*!: *integrable-in-subalg*)
then show $AE\ x\ in\ M.$ *cond-exp* $M\ F\ f\ x = g\ x$ **using** *AE-restr-to-subalg*[*OF subalg*] **by** *simp*
qed

lemma *cond-exp-F-meas*[*intro*, *simp*]:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*, *banach*}
assumes *integrable* $M\ f$
$f \in$ *borel-measurable* F
shows $AE\ x\ in\ M.$ *cond-exp* $M\ F\ f\ x = f\ x$
by (*rule has-cond-exp-charact*(*2*), *auto intro*: *assms*)

Congruence

lemma *has-cond-exp-cong*:
assumes *integrable* $M\ f$ $\bigwedge x.\ x \in$ *space* $M \Longrightarrow f\ x = g\ x$ *has-cond-exp* $M\ F\ g\ h$
shows *has-cond-exp* $M\ F\ f\ h$
proof (*intro has-cond-expI'*[*OF - assms*(*1*)], *goal-cases*)
case (*1 A*)
hence *set-lebesgue-integral* $M\ A\ f$ = *set-lebesgue-integral* $M\ A\ g$ **by** (*intro set-lebesgue-integral-cong*) (*meson assms*(*2*) *subalg in-mono subalgebra-def sets.sets-into-space subalgebra-def subsetD*)+
then show *?case* **using** *1 assms*(*3*) **by** (*simp add*: *has-cond-exp-def*)
qed (*auto simp add*: *has-cond-expD*[*OF assms*(*3*)])

lemma *cond-exp-cong*:

fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes *integrable M f integrable M g* $\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x = g\ x$
shows *AE x in M. cond-exp M F f x = cond-exp M F g x*
proof (*cases* $\exists h$. *has-cond-exp M F f h*)
case *True*
then obtain *h* **where** *h*: *has-cond-exp M F f h has-cond-exp M F g h* **using** *has-cond-exp-cong assms* **by** *metis*
show *?thesis* **using** *h*[*THEN has-cond-exp-charact*(*2*)] **by** *fastforce*
next
case *False*
moreover have $\nexists h$. *has-cond-exp M F g h* **using** *False has-cond-exp-cong assms* **by** *auto*
ultimately show *?thesis* **unfolding** *cond-exp-def* **by** *auto*
qed

lemma *has-cond-exp-cong-AE*:
assumes *integrable M f AE x in M. f x = g x has-cond-exp M F g h*
shows *has-cond-exp M F f h*
using *assms*(*1*,*2*) *subalg subalgebra-def subset-iff*
by (*intro has-cond-expI'*, *subst set-lebesgue-integral-cong-AE*[*OF - assms*(*1*)[*THEN borel-measurable-integrable*] *borel-measurable-integrable*(*1*)[*OF has-cond-expD*(*2*)[*OF assms*(*3*)]]]])
(*fast intro*: *has-cond-expD*[*OF assms*(*3*)] *integrable-cong-AE-imp*[*OF - - AE-symmetric*])+

lemma *has-cond-exp-cong-AE'*:
assumes $h \in$ *borel-measurable F AE x in M. h x = h' x has-cond-exp M F f h'*
shows *has-cond-exp M F f h*
using *assms*(*1*, *2*) *subalg subalgebra-def subset-iff*
using *AE-restr-to-subalg2*[*OF subalg assms*(*2*)] *measurable-from-subalg*
by (*intro has-cond-expI'* , *subst set-lebesgue-integral-cong-AE*[*OF - measurable-from-subalg*(*1*,*1*)[*OF subalg*], *OF - assms*(*1*) *has-cond-expD*(*4*)[*OF assms*(*3*)]]])
(*fast intro*: *has-cond-expD*[*OF assms*(*3*)] *integrable-cong-AE-imp*[*OF - - AE-symmetric*])+

lemma *cond-exp-cong-AE*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes *integrable M f integrable M g AE x in M. f x = g x*
shows *AE x in M. cond-exp M F f x = cond-exp M F g x*
proof (*cases* $\exists h$. *has-cond-exp M F f h*)
case *True*
then obtain *h* **where** *h*: *has-cond-exp M F f h has-cond-exp M F g h* **using** *has-cond-exp-cong-AE assms* **by** (*metis* (*mono-tags*, *lifting*) *eventually-mono*)
show *?thesis* **using** *h*[*THEN has-cond-exp-charact*(*2*)] **by** *fastforce*
next
case *False*
moreover have $\nexists h$. *has-cond-exp M F g h* **using** *False has-cond-exp-cong-AE assms* **by** *auto*
ultimately show *?thesis* **unfolding** *cond-exp-def* **by** *auto*
qed

lemma *has-cond-exp-real*:
fixes f :: $'a \Rightarrow$ *real*
assumes *integrable* M f
shows *has-cond-exp* M F f (*real-cond-exp* M F f)
by (*standard*, *auto intro*!: *real-cond-exp-intA assms*)

lemma *cond-exp-real*[*intro*]:
fixes f :: $'a \Rightarrow$ *real*
assumes *integrable* M f
shows *AE* x *in* M. *cond-exp* M F f x = *real-cond-exp* M F f x
using *has-cond-exp-charact has-cond-exp-real assms* **by** *blast*

lemma *cond-exp-cmult*:
fixes f :: $'a \Rightarrow$ *real*
assumes *integrable* M f
shows *AE* x *in* M. *cond-exp* M F (λx. $c * f\ x$) x = c $*$ *cond-exp* M F f x
using *real-cond-exp-cmult*[*OF assms*(*1*), *of c*] *assms*(*1*)[*THEN cond-exp-real*] *assms*(*1*)[*THEN integrable-mult-right*, *THEN cond-exp-real*, *of c*] **by** *fastforce*

Indicator functions

lemma *has-cond-exp-indicator*:
assumes $A \in$ *sets* M *emeasure* M $A < \infty$
shows *has-cond-exp* M F (λx. *indicat-real* A x $*_R$ y) (λx. *real-cond-exp* M F (*indicator* A) x $*_R$ y)
proof (*intro has-cond-expI'*, *goal-cases*)
case (*1 B*)
have $\int x\in B$. (*indicat-real* A x $*_R$ y) ∂M = ($\int x\in B$. *indicat-real* A x ∂M) $*_R$ y **using** *assms* **by** (*intro set-integral-scaleR-left*, *meson 1 in-mono subalg subalgebra-def*, *blast*)
also have ... = ($\int x\in B$. *real-cond-exp* M F (*indicator* A) x ∂M) $*_R$ y **using** *1 assms* **by** (*subst real-cond-exp-intA*, *auto*)
also have ... = $\int x\in B$. (*real-cond-exp* M F (*indicator* A) x $*_R$ y) ∂M **using** *assms* **by** (*intro set-integral-scaleR-left*[*symmetric*], *meson 1 in-mono subalg subalgebra-def*, *blast*)
finally show *?case* **.**
next
case *2*
then show *?case* **using** *integrable-scaleR-left integrable-real-indicator assms* **by** *blast*
next
case *3*
show *?case* **using** *assms* **by** (*intro integrable-scaleR-left*, *intro real-cond-exp-int*, *blast*+)
next
case *4*
then show *?case* **by** (*intro borel-measurable-scaleR*, *intro Conditional-Expectation.borel-measurable-cond-exp*, *simp*)
qed

lemma *cond-exp-indicator*[*intro*]:
fixes y :: $'b$::{*second-countable-topology*,*banach*}
assumes [*measurable*]: $A \in$ *sets M emeasure M A* $< \infty$
shows *AE x in M*. *cond-exp M F* (λx. *indicat-real A x* $*_R$ y) x = *cond-exp M F* (*indicator A*) x $*_R$ y
proof −
have *AE x in M*. *cond-exp M F* (λx. *indicat-real A x* $*_R$ y) x = *real-cond-exp M F* (*indicator A*) x $*_R$ y **using** *has-cond-exp-indicator*[*OF assms*] *has-cond-exp-charact* **by** *blast*
thus *?thesis* **using** *cond-exp-real*[*OF integrable-real-indicator*, *OF assms*] **by** *fast-force*
qed

Addition

lemma *has-cond-exp-add*:
fixes f g :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes *has-cond-exp M F f f'* *has-cond-exp M F g g'*
shows *has-cond-exp M F* (λx. f x + g x) (λx. f' x + g' x)
proof (*intro has-cond-expI'*, *goal-cases*)
case (*1 A*)
have $\int x{\in}A.\ (f\ x\ +\ g\ x)\partial M$ = ($\int x{\in}A.\ f\ x\ \partial M$) + ($\int x{\in}A.\ g\ x\ \partial M$) **using** *assms*[*THEN has-cond-expD*(*2*)] *subalg 1* **by** (*intro set-integral-add*(*2*), *auto simp add*: *subalgebra-def set-integrable-def intro*: *integrable-mult-indicator*)
also have ... = ($\int x{\in}A.\ f'\ x\ \partial M$) + ($\int x{\in}A.\ g'\ x\ \partial M$) **using** *assms*[*THEN has-cond-expD*(*1*)[*OF - 1*]] **by** *argo*
also have ... = $\int x{\in}A.\ (f'\ x\ +\ g'\ x)\partial M$ **using** *assms*[*THEN has-cond-expD*(*3*)] *subalg 1* **by** (*intro set-integral-add*(*2*)[*symmetric*], *auto simp add*: *subalgebra-def set-integrable-def intro*: *integrable-mult-indicator*)
finally show *?case* **.**
next
case *2*
then show *?case* **by** (*metis Bochner-Integration.integrable-add assms has-cond-expD*(*2*))
next
case *3*
then show *?case* **by** (*metis Bochner-Integration.integrable-add assms has-cond-expD*(*3*))
next
case *4*
then show *?case* **using** *assms borel-measurable-add has-cond-expD*(*4*) **by** *blast*
qed

lemma *has-cond-exp-scaleR-right*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes *has-cond-exp M F f f'*
shows *has-cond-exp M F* (λx. c $*_R$ f x) (λx. c $*_R$ f' x)
using *has-cond-expD*[*OF assms*] **by** (*intro has-cond-expI'*, *auto*)

lemma *cond-exp-scaleR-right*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes *integrable M f*

shows *AE x in M. cond-exp M F* ($\lambda x.\ c *_R f\ x$) $x = c *_R$ *cond-exp M F f x*
proof (*cases* $\exists f'$. *has-cond-exp M F f f'*)
case *True*
then show *?thesis* **using** *assms has-cond-exp-charact has-cond-exp-scaleR-right*
by *metis*
next
case *False*
show *?thesis*
proof (*cases c* = *0*)
case *True*
then show *?thesis* **by** *simp*
next
case *c-nonzero*: *False*
have $\nexists f'$. *has-cond-exp M F* ($\lambda x.\ c *_R f\ x$) f'
proof (*standard*, *goal-cases*)
case *1*
then obtain f' **where** f': *has-cond-exp M F* ($\lambda x.\ c *_R f\ x$) f' **by** *blast*
have *has-cond-exp M F f* ($\lambda x.$ *inverse* $c *_R f'\ x$) **using** *has-cond-expD*[*OF*
f'] *divideR-right*[*OF c-nonzero*] *assms* **by** (*intro has-cond-expI'*, *auto*)
then show *?case* **using** *False* **by** *blast*
qed
then show *?thesis* **using** *cond-exp-null*[*OF False*] *cond-exp-null* **by** *force*
qed
qed

lemma *cond-exp-uminus*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *integrable M f*
shows *AE x in M. cond-exp M F* ($\lambda x.\ -f\ x$) $x = -$ *cond-exp M F f x*
using *cond-exp-scaleR-right*[*OF assms*, *of* −*1*] **by** *force*

lemma *has-cond-exp-simple*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *simple-function M f emeasure M* $\{y \in$ *space M*. $f\ y \neq 0\} \neq \infty$
shows *has-cond-exp M F f* (*cond-exp M F f*)
using *assms*
proof (*induction rule*: *simple-integrable-function-induct*)
case (*cong f g*)
then show *?case* **using** *has-cond-exp-cong* **by** (*metis* (*no-types*, *opaque-lifting*)
Bochner-Integration.integrable-cong has-cond-expD(*2*) *has-cond-exp-charact*(*1*))
next
case (*indicator A y*)
then show *?case* **using** *has-cond-exp-charact*[*OF has-cond-exp-indicator*] **by** *fast*
next
case (*add u v*)
then show *?case* **using** *has-cond-exp-add has-cond-exp-charact*(*1*) **by** *blast*
qed

lemma *cond-exp-contraction-real*:

fixes f :: $'a \Rightarrow$ *real*
assumes *integrable*[*measurable*]: *integrable M f*
shows *AE x in M. norm* (*cond-exp M F f x*) $\leq$ *cond-exp M F* (λx. *norm* (*f x*)) *x*
proof−
have *int*: *integrable M* (λx. *norm* (*f x*)) **using** *assms* **by** *blast*
have ∗: *AE x in M*. *0* $\leq$ *cond-exp M F* (λx. *norm* (*f x*)) *x* **using** *cond-exp-real*[*THEN AE-symmetric*, *OF integrable-norm*[*OF integrable*]] *real-cond-exp-ge-c*[*OF integrable-norm*[*OF integrable*], *of 0*] *norm-ge-zero* **by** *fastforce*
have ∗∗: $A \in$ *sets F* $\Longrightarrow \int x{\in}A.\ |f\ x|\ \partial M = \int x{\in}A.$ *real-cond-exp M F* (λx. *norm* (*f x*)) *x* ∂M **for** *A* **unfolding** *real-norm-def* **using** *assms integrable-abs real-cond-exp-intA* **by** *blast*

have *norm-int*: $A \in$ *sets F* $\Longrightarrow (\int x{\in}A.\ |f\ x|\ \partial M) = (\int^+ x{\in}A.\ |f\ x|\ \partial M)$ **for** *A* **using** *assms* **by** (*intro nn-set-integral-eq-set-integral*[*symmetric*], *blast*, *fastforce*) (*meson subalg subalgebra-def subsetD*)

have *AE x in M*. *real-cond-exp M F* (λx. *norm* (*f x*)) *x* $\geq$ *0* **using** *int real-cond-exp-ge-c* **by** *force*
hence *cond-exp-norm-int*: $A \in$ *sets F* $\Longrightarrow (\int x{\in}A.$ *real-cond-exp M F* (λx. *norm* (*f x*)) *x* $\partial M) = (\int^+ x{\in}A.$ *real-cond-exp M F* (λx. *norm* (*f x*)) *x* $\partial M)$ **for** *A* **using** *assms* **by** (*intro nn-set-integral-eq-set-integral*[*symmetric*], *blast*, *fastforce*) (*meson subalg subalgebra-def subsetD*)

have $A \in$ *sets F* $\Longrightarrow \int^+ x{\in}A.\ |f\ x| \partial M = \int^+ x{\in}A.$ *real-cond-exp M F* (λx. *norm* (*f x*)) *x* ∂M **for** *A* **using** ∗∗ *norm-int cond-exp-norm-int* **by** (*auto simp add*: *nn-integral-set-ennreal*)
moreover have (λx. *ennreal* $|f\ x|$) $\in$ *borel-measurable M* **by** *measurable*
moreover have (λx. *ennreal* (*real-cond-exp M F* (λx. *norm* (*f x*)) *x*)) $\in$ *borel-measurable F* **by** *measurable*
ultimately have *AE x in M*. *nn-cond-exp M F* (λx. *ennreal* $|f\ x|$) *x* = *real-cond-exp M F* (λx. *norm* (*f x*)) *x* **by** (*intro nn-cond-exp-charact*[*THEN AE-symmetric*], *auto*)
hence *AE x in M*. *nn-cond-exp M F* (λx. *ennreal* $|f\ x|$) *x* $\leq$ *cond-exp M F* (λx. *norm* (*f x*)) *x* **using** *cond-exp-real*[*OF int*] **by** *force*
moreover have *AE x in M*. |*real-cond-exp M F f x*| = *norm* (*cond-exp M F f x*) **unfolding** *real-norm-def* **using** *cond-exp-real*[*OF assms*] ∗ **by** *force*
ultimately have *AE x in M*. *ennreal* (*norm* (*cond-exp M F f x*)) $\leq$ *cond-exp M F* (λx. *norm* (*f x*)) *x* **using** *real-cond-exp-abs*[*OF assms*[*THEN borel-measurable-integrable*]] **by** *fastforce*
hence *AE x in M*. *enn2real* (*ennreal* (*norm* (*cond-exp M F f x*))) $\leq$ *enn2real* (*cond-exp M F* (λx. *norm* (*f x*)) *x*) **using** *ennreal-le-iff2* **by** *force*
thus *?thesis* **using** ∗ **by** *fastforce*
qed

lemma *cond-exp-contraction-simple*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*, *banach*}
assumes *simple-function M f emeasure M* $\{y \in$ *space M*. *f y* $\neq 0\} \neq \infty$
shows *AE x in M. norm* (*cond-exp M F f x*) $\leq$ *cond-exp M F* (λx. *norm* (*f x*)) *x*
using *assms*

proof (*induction rule*: *simple-integrable-function-induct*)
case (*cong f g*)
hence *ae*: *AE x in M. f x = g x* **by** *blast*
hence *AE x in M. cond-exp M F f x = cond-exp M F g x* **using** *cong has-cond-exp-simple* **by** (*subst cond-exp-cong-AE*) (*auto intro*!: *has-cond-expD*(*2*))
hence *AE x in M. norm* (*cond-exp M F f x*) = *norm* (*cond-exp M F g x*) **by** *force*
moreover have *AE x in M. cond-exp M F* (*λx. norm* (*f x*)) *x = cond-exp M F* (*λx. norm* (*g x*)) *x* **using** *ae cong has-cond-exp-simple* **by** (*subst cond-exp-cong-AE*) (*auto dest*: *has-cond-expD*)
ultimately show *?case* **using** *cong*(*6*) **by** *fastforce*
next
case (*indicator A y*)
hence *AE x in M. cond-exp M F* (*λa. indicator A a* $*_R$ *y*) *x = cond-exp M F* (*indicator A*) *x* $*_R$ *y* **by** *blast*
hence ∗: *AE x in M. norm* (*cond-exp M F* (*λa. indicat-real A a* $*_R$ *y*) *x*) ≤ *norm y* ∗ *cond-exp M F* (*λx. norm* (*indicat-real A x*)) *x* **using** *cond-exp-contraction-real*[*OF integrable-real-indicator*, *OF indicator*] **by** *fastforce*

have *AE x in M. norm y* ∗ *cond-exp M F* (*λx. norm* (*indicat-real A x*)) *x = norm y* ∗ *real-cond-exp M F* (*λx. norm* (*indicat-real A x*)) *x* **using** *cond-exp-real*[*OF integrable-real-indicator*, *OF indicator*] **by** *fastforce*
moreover have *AE x in M. cond-exp M F* (*λx. norm y* ∗ *norm* (*indicat-real A x*)) *x = real-cond-exp M F* (*λx. norm y* ∗ *norm* (*indicat-real A x*)) *x* **using** *indicator* **by** (*intro cond-exp-real*, *auto*)
ultimately have *AE x in M. norm y* ∗ *cond-exp M F* (*λx. norm* (*indicat-real A x*)) *x = cond-exp M F* (*λx. norm y* ∗ *norm* (*indicat-real A x*)) *x* **using** *real-cond-exp-cmult*[*of λx. norm* (*indicat-real A x*) *norm y*] *indicator* **by** *fastforce*
moreover have (*λx. norm y* ∗ *norm* (*indicat-real A x*)) = (*λx. norm* (*indicat-real A x* $*_R$ *y*)) **by** *force*
ultimately show *?case* **using** ∗ **by** *force*
next
case (*add u v*)
have *AE x in M. norm* (*cond-exp M F* (*λa. u a + v a*) *x*) = *norm* (*cond-exp M F u x + cond-exp M F v x*) **using** *has-cond-exp-charact*(*2*)[*OF has-cond-exp-add*, *OF has-cond-exp-simple*(*1*,*1*), *OF add*(*1*,*2*,*3*,*4*)] **by** *fastforce*
moreover have *AE x in M. norm* (*cond-exp M F u x + cond-exp M F v x*) ≤ *norm* (*cond-exp M F u x*) + *norm* (*cond-exp M F v x*) **using** *norm-triangle-ineq* **by** *blast*
moreover have *AE x in M. norm* (*cond-exp M F u x*) + *norm* (*cond-exp M F v x*) ≤ *cond-exp M F* (*λx. norm* (*u x*)) *x + cond-exp M F* (*λx. norm* (*v x*)) *x* **using** *add*(*6*,*7*) **by** *fastforce*
moreover have *AE x in M. cond-exp M F* (*λx. norm* (*u x*)) *x + cond-exp M F* (*λx. norm* (*v x*)) *x = cond-exp M F* (*λx. norm* (*u x*) + *norm* (*v x*)) *x* **using** *integrable-simple-function*[*OF add*(*1*,*2*)] *integrable-simple-function*[*OF add*(*3*,*4*)] **by** (*intro has-cond-exp-charact*(*2*)[*OF has-cond-exp-add*[*OF has-cond-exp-charact*(*1*,*1*)], *THEN AE-symmetric*], *auto intro*: *has-cond-exp-real*)
moreover have *AE x in M. cond-exp M F* (*λx. norm* (*u x*) + *norm* (*v x*)) *x = cond-exp M F* (*λx. norm* (*u x + v x*)) *x* **using** *add*(*5*) *integrable-simple-function*[*OF*

add(*1*,*2*)] *integrable-simple-function*[*OF add*(*3*,*4*)] **by** (*intro cond-exp-cong*, *auto*)
ultimately show *?case* **by** *force*
qed

lemma *has-cond-exp-lim*:
fixes $f :: 'a \Rightarrow 'b$::{*second-countable-topology*, *banach*}
assumes *integrable*[*measurable*]: *integrable M f*
and $\bigwedge i.$ *simple-function M* (*s i*)
and $\bigwedge i.$ *emeasure M* $\{y \in$ *space M*. *s i y* $\neq 0\} \neq \infty$
and $\bigwedge x.$ $x \in$ *space M* $\Longrightarrow$ ($\lambda i.$ *s i x*) $\longrightarrow$ *f x*
and $\bigwedge x\ i.$ $x \in$ *space M* $\Longrightarrow$ *norm* (*s i x*) $\leq$ *2* $*$ *norm* (*f x*)
obtains *r*
where *has-cond-exp M F f* ($\lambda x.$ *lim* ($\lambda i.$ *cond-exp M F* (*s* (*r i*)) *x*))
AE x in M. *convergent* ($\lambda i.$ *cond-exp M F* (*s* (*r i*)) *x*)
strict-mono r
proof −
have [*measurable*]: (*s i*) $\in$ *borel-measurable M* **for** *i* **using** *assms*(*2*) **by** (*simp add*: *borel-measurable-simple-function*)
have *integrable-s*: *integrable M* ($\lambda x.$ *s i x*) **for** *i* **using** *assms*(*2*) *assms*(*3*) *integrable-simple-function* **by** *blast*
have *integrable-4f*: *integrable M* ($\lambda x.$ *4* $*$ *norm* (*f x*)) **using** *assms*(*1*) **by** *simp*
have *integrable-2f*: *integrable M* ($\lambda x.$ *2* $*$ *norm* (*f x*)) **using** *assms*(*1*) **by** *simp*
have *integrable-2-cond-exp-norm-f*: *integrable M* ($\lambda x.$ *2* $*$ *cond-exp M F* ($\lambda x.$ *norm* (*f x*)) *x*) **by** *fast*

have *emeasure M* $\{y \in$ *space M*. *s i y* $-$ *s j y* $\neq 0\} \leq$ *emeasure M* $\{y \in$ *space M*. *s i y* $\neq 0\}$ $+$ *emeasure M* $\{y \in$ *space M*. *s j y* $\neq 0\}$ **for** *i j* **using** *simple-functionD*(*2*)[*OF assms*(*2*)] **by** (*intro order-trans*[*OF emeasure-mono emeasure-subadditive*], *auto*)
hence *fin-sup*: *emeasure M* $\{y \in$ *space M*. *s i y* $-$ *s j y* $\neq 0\} \neq \infty$ **for** *i j* **using** *assms*(*3*) **by** (*metis* (*mono-tags*) *ennreal-add-eq-top linorder-not-less top.not-eq-extremum infinity-ennreal-def*)

have *emeasure M* $\{y \in$ *space M*. *norm* (*s i y* $-$ *s j y*) $\neq 0\} \leq$ *emeasure M* $\{y \in$ *space M*. *s i y* $\neq 0\}$ $+$ *emeasure M* $\{y \in$ *space M*. *s j y* $\neq 0\}$ **for** *i j* **using** *simple-functionD*(*2*)[*OF assms*(*2*)] **by** (*intro order-trans*[*OF emeasure-mono emeasure-subadditive*], *auto*)
hence *fin-sup-norm*: *emeasure M* $\{y \in$ *space M*. *norm* (*s i y* $-$ *s j y*) $\neq 0\} \neq \infty$ **for** *i j* **using** *assms*(*3*) **by** (*metis* (*mono-tags*) *ennreal-add-eq-top linorder-not-less top.not-eq-extremum infinity-ennreal-def*)

have *Cauchy*: *Cauchy* ($\lambda n.$ *s n x*) **if** $x \in$ *space M* **for** *x* **using** *assms*(*4*) *LIMSEQ-imp-Cauchy that* **by** *blast*
hence *bounded-range-s*: *bounded* (*range* ($\lambda n.$ *s n x*)) **if** $x \in$ *space M* **for** *x* **using** *that cauchy-imp-bounded* **by** *fast*

have *AE x in M*. ($\lambda n.$ *diameter* $\{$*s i x* $|$ *i*. $n \leq i\}$) $\longrightarrow$ *0* **using** *Cauchy cauchy-iff-diameter-tends-to-zero-and-bounded* **by** *fast*
moreover have ($\lambda x.$ *diameter* $\{$*s i x* $|$*i*. $n \leq i\}$) $\in$ *borel-measurable M* **for** *n*

using *bounded-range-s borel-measurable-diameter* **by** *measurable*
moreover have *AE x in M. norm* (*diameter* $\{s\ i\ x\ |i.\ n \le i\}$) $\le$ *4* $*$ *norm* (*f x*) **for** *n*
proof –
{
fix *x* **assume** *x*: $x \in$ *space M*
have *diameter* $\{s\ i\ x\ |i.\ n \le i\} \le 2 * norm\ (f\ x) + 2 * norm\ (f\ x)$ **by** (*intro diameter-le*, *blast*, *subst dist-norm*[*symmetric*], *intro dist-triangle3*[*THEN order-trans*, *of 0*], *intro add-mono*) (*auto intro*: *assms*(*5*)[*OF x*])
hence *norm* (*diameter* $\{s\ i\ x\ |i.\ n \le i\}$) $\le$ *4* $*$ *norm* (*f x*) **using** *diameter-ge-0*[*OF bounded-subset*[*OF bounded-range-s*], *OF x*, *of* $\{s\ i\ x\ |i.\ n \le i\}$] **by** *force*
}
thus *?thesis* **by** *fast*
qed
ultimately have *diameter-tendsto-zero*: (λn. *LINT x|M. diameter* $\{s\ i\ x\ |\ i.\ n \le i\}$) $\longrightarrow$ *0* **by** (*intro integral-dominated-convergence*[*OF borel-measurable-const*[*of 0*] - *integrable-4f*, *simplified*]) (*fast*+)

have *diameter-integrable*: *integrable M* (λx. *diameter* $\{s\ i\ x\ |\ i.\ n \le i\}$) **for** *n* **using** *assms*(*1*,*5*) **by** (*intro integrable-bound-diameter*[*OF bounded-range-s integrable-2f*], *auto*)

have *dist-integrable*: *integrable M* (λx. *dist* (*s i x*) (*s j x*)) **for** *i j*
using *assms*(*5*) *dist-triangle3*[*of s i - - 0*, *THEN order-trans*, *OF add-mono*, *of - 2* $*$ *norm* (*f* -)]
by (*intro Bochner-Integration.integrable-bound*[*OF integrable-4f*]) *fastforce*+

hence *dist-norm-integrable*: *integrable M* (λx. *norm* (*s i x* $-$ *s j x*)) **for** *i j* **unfolding** *dist-norm* **by** *presburger*

have $\exists N.\ \forall i \ge N.\ \forall j \ge N.$ *LINT x|M. dist* (*cond-exp M F* (*s i*) *x*) (*cond-exp M F* (*s j*) *x*) $<$ *e* **if** *e-pos*: $e > 0$ **for** *e*
proof –
obtain *N* **where** $*$: *LINT x|M. diameter* $\{s\ i\ x\ |\ i.\ n \le i\} < e$ **if** $n \ge N$ **for** *n* **using** *that order-tendsto-iff*[*THEN iffD1*, *OF diameter-tendsto-zero*, *unfolded eventually-sequentially*] *e-pos* **by** *presburger*
{
fix *i j x* **assume** *asm*: $i \ge N$ $j \ge N$ $x \in$ *space M*
have *case-prod dist* ` ($\{s\ i\ x\ |i.\ N \le i\} \times \{s\ i\ x\ |i.\ N \le i\}$) = *case-prod* ($\lambda i\ j$. *dist* (*s i x*) (*s j x*)) ` ($\{N..\} \times \{N..\}$) **by** *fast*
hence *diameter* $\{s\ i\ x\ |\ i.\ N \le i\}$ = (*SUP* (*i*, *j*) $\in \{N..\} \times \{N..\}$. *dist* (*s i x*) (*s j x*)) **unfolding** *diameter-def* **by** *auto*
moreover have (*SUP* (*i*, *j*) $\in \{N..\} \times \{N..\}$. *dist* (*s i x*) (*s j x*)) $\ge$ *dist* (*s i x*) (*s j x*) **using** *asm bounded-imp-bdd-above*[*OF bounded-imp-dist-bounded*, *OF bounded-range-s*] **by** (*intro cSup-upper*, *auto*)
ultimately have *diameter* $\{s\ i\ x\ |\ i.\ N \le i\} \ge$ *dist* (*s i x*) (*s j x*) **by** *presburger*
}

hence *LINT x|M. dist (s i x) (s j x)* $<$ *e* **if** *i* $\geq$ *N j* $\geq$ *N* **for** *i j* **using** *that* $*$ **by** (*intro integral-mono*[*OF dist-integrable diameter-integrable*, *THEN order.strict-trans1*], *blast+*)
moreover have *LINT x|M. dist (cond-exp M F (s i) x) (cond-exp M F (s j) x)* $\leq$ *LINT x|M. dist (s i x) (s j x)* **for** *i j*
proof−
have *LINT x|M. dist (cond-exp M F (s i) x) (cond-exp M F (s j) x)* = *LINT x|M. norm (cond-exp M F (s i) x* + − *1* $*_R$ *cond-exp M F (s j) x)* **unfolding** *dist-norm* **by** *simp*
also have ... = *LINT x|M. norm (cond-exp M F* (λ*x. s i x* − *s j x) x)* **using** *has-cond-exp-charact(2)*[*OF has-cond-exp-add*[*OF - has-cond-exp-scaleR-right*, *OF has-cond-exp-charact(1,1)*, *OF has-cond-exp-simple(1,1)*[*OF assms(2,3)*]], *THEN AE-symmetric*, *of i −1 j*] **by** (*intro integral-cong-AE*) *force+*
also have ... $\leq$ *LINT x|M. cond-exp M F* (λ*x. norm (s i x* − *s j x)) x* **using** *cond-exp-contraction-simple*[*OF - fin-sup*, *of i j*] *integrable-cond-exp assms(2)* **by** (*intro integral-mono-AE*, *fast+*)
also have ... = *LINT x|M. norm (s i x* − *s j x)* **unfolding** *set-integral-space(1)*[*OF integrable-cond-exp*, *symmetric*] *set-integral-space*[*OF dist-norm-integrable*, *symmetric*] **by** (*intro has-cond-expD(1)*[*OF has-cond-exp-simple*[*OF - fin-sup-norm*], *symmetric*]) (*metis assms(2) simple-function-compose1 simple-function-diff*, *metis sets.top subalg subalgebra-def*)
finally show *?thesis* **unfolding** *dist-norm* **.**
qed
ultimately show *?thesis* **using** *order.strict-trans1* **by** *meson*
qed
then obtain *r* **where** *strict-mono-r*: *strict-mono r* **and** *AE-Cauchy*: *AE x in M. Cauchy* (λ*i. cond-exp M F (s (r i)) x)* **by** (*rule cauchy-L1-AE-cauchy-subseq*[*OF integrable-cond-exp*], *auto*)
hence *ae-lim-cond-exp*: *AE x in M.* (λ*n. cond-exp M F (s (r n)) x)* $\longrightarrow$ *lim* (λ*n. cond-exp M F (s (r n)) x)* **using** *Cauchy-convergent-iff convergent-LIMSEQ-iff* **by** *fastforce*

have *cond-exp-bounded*: *AE x in M. norm (cond-exp M F (s (r n)) x)* $\leq$ *cond-exp M F* (λ*x. 2* $*$ *norm (f x)) x* **for** *n*
proof −
have *AE x in M. norm (cond-exp M F (s (r n)) x)* $\leq$ *cond-exp M F* (λ*x. norm (s (r n) x)) x* **by** (*rule cond-exp-contraction-simple*[*OF assms(2,3)*])
moreover have *AE x in M. real-cond-exp M F* (λ*x. norm (s (r n) x)) x* $\leq$ *real-cond-exp M F* (λ*x. 2* $*$ *norm (f x)) x* **using** *integrable-s integrable-2f assms(5)* **by** (*intro real-cond-exp-mono*, *auto*)
ultimately show *?thesis* **using** *cond-exp-real*[*OF integrable-norm*, *OF integrable-s*, *of r n*] *cond-exp-real*[*OF integrable-2f*] **by** *force*
qed
have *lim-integrable*: *integrable M* (λ*x. lim* (λ*i. cond-exp M F (s (r i)) x))* **by** (*intro integrable-dominated-convergence*[*OF - borel-measurable-cond-exp' integrable-cond-exp ae-lim-cond-exp cond-exp-bounded*], *simp*)

{
fix *A* **assume** *A-in-sets-F*: *A* $\in$ *sets F*

have *AE x in M. norm (indicator A x* $*_R$ *cond-exp M F (s (r n)) x)* $\leq$ *cond-exp M F (λx. 2 * norm (f x)) x* **for** *n*
proof −
have *AE x in M. norm (indicator A x* $*_R$ *cond-exp M F (s (r n)) x)* $\leq$ *norm (cond-exp M F (s (r n)) x)* **unfolding** *indicator-def* **by** *simp*
thus *?thesis* **using** *cond-exp-bounded[of n]* **by** *force*
qed
hence *lim-cond-exp-int: (λn. LINT x:A|M. cond-exp M F (s (r n)) x)* $\longrightarrow$ *LINT x:A|M. lim (λn. cond-exp M F (s (r n)) x)*
using *ae-lim-cond-exp measurable-from-subalg[OF subalg borel-measurable-indicator, OF A-in-sets-F] cond-exp-bounded*
unfolding *set-lebesgue-integral-def*
by *(intro integral-dominated-convergence[OF borel-measurable-scaleR borel-measurable-scaleR integrable-cond-exp]) (fastforce simp add: tendsto-scaleR)+*

have *AE x in M. norm (indicator A x* $*_R$ *s (r n) x)* $\leq$ *2 * norm (f x)* **for** *n*
proof −
have *AE x in M. norm (indicator A x* $*_R$ *s (r n) x)* $\leq$ *norm (s (r n) x)* **unfolding** *indicator-def* **by** *simp*
thus *?thesis* **using** *assms(5)[of - r n]* **by** *fastforce*
qed
hence *lim-s-int: (λn. LINT x:A|M. s (r n) x)* $\longrightarrow$ *LINT x:A|M. f x*
using *measurable-from-subalg[OF subalg borel-measurable-indicator, OF A-in-sets-F] LIMSEQ-subseq-LIMSEQ[OF assms(4) strict-mono-r] assms(5)*
unfolding *set-lebesgue-integral-def comp-def*
by *(intro integral-dominated-convergence[OF borel-measurable-scaleR borel-measurable-scaleR integrable-2f]) (fastforce simp add: tendsto-scaleR)+*

have *LINT x:A|M. lim (λn. cond-exp M F (s (r n)) x) = lim (λn. LINT x:A|M. cond-exp M F (s (r n)) x)* **using** *limI[OF lim-cond-exp-int]* **by** *argo*
also have ... = *lim (λn. LINT x:A|M. s (r n) x)* **using** *has-cond-expD(1)[OF has-cond-exp-simple[OF assms(2,3)] A-in-sets-F, symmetric]* **by** *presburger*
also have ... = *LINT x:A|M. f x* **using** *limI[OF lim-s-int]* **by** *argo*
finally have *LINT x:A|M. lim (λn. cond-exp M F (s (r n)) x) = LINT x:A|M. f x* .
}
hence *has-cond-exp M F f (λx. lim (λi. cond-exp M F (s (r i)) x))* **using** *assms(1) lim-integrable* **by** *(intro has-cond-expI′, auto)*
thus *thesis* **using** *AE-Cauchy Cauchy-convergent strict-mono-r* **by** *(auto intro!: that)*
qed

lemma *cond-exp-lim*:
fixes *f :: ′a ⇒ ′b::{second-countable-topology, banach}*
assumes *[measurable]:integrable M f*
and $\bigwedge$*i. simple-function M (s i)*
and $\bigwedge$*i. emeasure M {y ∈ space M. s i y ≠ 0} ≠ ∞*
and $\bigwedge$*x. x ∈ space M* $\Longrightarrow$ *(λi. s i x)* $\longrightarrow$ *f x*
and $\bigwedge$*x i. x ∈ space M* $\Longrightarrow$ *norm (s i x)* $\leq$ *2 * norm (f x)*

obtains *r* **where** *AE x in M*. (*λi. cond-exp M F* (*s* (*r i*)) *x*) ⟶ *cond-exp M F f x strict-mono r*
proof –
obtain *r* **where** *AE x in M*. *cond-exp M F f x* = *lim* (*λi. cond-exp M F* (*s* (*r i*)) *x*) *AE x in M*. *convergent* (*λi. cond-exp M F* (*s* (*r i*)) *x*) *strict-mono r* **using** *has-cond-exp-charact*(*2*) **by** (*auto intro*: *has-cond-exp-lim*[*OF assms*])
thus *?thesis* **by** (*auto intro*!: *that*[*of r*] *simp*: *convergent-LIMSEQ-iff*)
qed

lemma *has-cond-expI*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *integrable M f*
shows *has-cond-exp M F f* (*cond-exp M F f*)
using *assms*
proof (*induction rule*: *integrable-induct'*)
case (*base A c*)
show *?case* **using** *has-cond-exp-indicator*[*OF base*(*1*,*2*)] *has-cond-exp-charact*(*1*) **by** *blast*
next
case (*add u v*)
show *?case* **using** *has-cond-exp-add*[*OF add*(*3*,*4*)] *has-cond-exp-charact*(*1*) **by** *blast*
next
case (*lim f s*)
show *?case* **using** *has-cond-exp-lim*[*OF lim*(*1*,*3*,*4*,*5*,*6*)] *has-cond-exp-charact*(*1*) **by** *meson*
qed

lemma *has-cond-exp-nested-subalg*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*, *banach*}
assumes *subalgebra G F has-cond-exp M F f h has-cond-exp M G f h'*
shows *has-cond-exp M F h' h*
by *standard* (*metis assms has-cond-expD in-mono subalgebra-def*)+

lemma *cond-exp-nested-subalg*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *integrable M f subalgebra M G subalgebra G F*
shows *AE ξ in M*. *cond-exp M F f ξ* = *cond-exp M F* (*cond-exp M G f*) *ξ*
using *has-cond-expI assms sigma-finite-subalgebra-def* **by** (*auto intro*!: *has-cond-exp-nested-subalg*[*THEN has-cond-exp-charact*(*2*), *THEN AE-symmetric*] *sigma-finite-subalgebra.has-cond-expI*[*OF sigma-finite-subalgebra.intro*[*OF assms*(*2*)]] *nested-subalg-is-sigma-finite*)

lemma *cond-exp-set-integral*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *integrable M f A* ∈ *sets F*
shows (∫ *x* ∈ *A*. *f x* ∂*M*) = (∫ *x* ∈ *A*. *cond-exp M F f x* ∂*M*)
using *has-cond-expD*(*1*)[*OF has-cond-expI*, *OF assms*] **by** *argo*

lemma *cond-exp-add*:

fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *integrable M f integrable M g*
shows *AE x in M*. *cond-exp M F* (*λx*. *f x* + *g x*) *x* = *cond-exp M F f x* + *cond-exp M F g x*
using *has-cond-exp-add*[*OF has-cond-expI*(*1*,*1*), *OF assms*, *THEN has-cond-exp-charact*(*2*)]
.

lemma *cond-exp-diff*:
fixes *f* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*}
assumes *integrable M f integrable M g*
shows *AE x in M*. *cond-exp M F* (*λx*. *f x* − *g x*) *x* = *cond-exp M F f x* − *cond-exp M F g x*
using *has-cond-exp-add*[*OF - has-cond-exp-scaleR-right*, *OF has-cond-expI*(*1*,*1*), *OF assms*, *THEN has-cond-exp-charact*(*2*), *of* −*1*] **by** *simp*

lemma *cond-exp-diff'*:
fixes *f* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*}
assumes *integrable M f integrable M g*
shows *AE x in M*. *cond-exp M F* (*f* − *g*) *x* = *cond-exp M F f x* − *cond-exp M F g x*
unfolding *fun-diff-def* **using** *assms* **by** (*rule cond-exp-diff*)

lemma *cond-exp-contraction*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*, *banach*}
assumes *integrable M f*
shows *AE x in M*. *norm* (*cond-exp M F f x*) ≤ *cond-exp M F* (*λx*. *norm* (*f x*)) *x*
proof −
obtain *s* **where** *s*: ⋀*i*. *simple-function M* (*s i*) ⋀*i*. *emeasure M* {*y* ∈ *space M*. *s i y* ≠ *0*} ≠ ∞ ⋀*x*. *x* ∈ *space M* ⟹ (*λi*. *s i x*) ⟶ *f x* ⋀*i x*. *x* ∈ *space M* ⟹ *norm* (*s i x*) ≤ *2* ∗ *norm* (*f x*)
by (*blast intro*: *integrable-implies-simple-function-sequence*[*OF assms*])

obtain *r* **where** *r*: *AE x in M*. (*λi*. *cond-exp M F* (*s* (*r i*)) *x*) ⟶ *cond-exp M F f x strict-mono r* **using** *cond-exp-lim*[*OF assms s*] **by** *blast*

have *norm-s-r*: ⋀*i*. *simple-function M* (*λx*. *norm* (*s* (*r i*) *x*)) ⋀*i*. *emeasure M* {*y* ∈ *space M*. *norm* (*s* (*r i*) *y*) ≠ *0*} ≠ ∞ ⋀*x*. *x* ∈ *space M* ⟹ (*λi*. *norm* (*s* (*r i*) *x*)) ⟶ *norm* (*f x*) ⋀*i x*. *x* ∈ *space M* ⟹ *norm* (*norm* (*s* (*r i*) *x*)) ≤ *2* ∗ *norm* (*norm* (*f x*))
using *s* **by** (*auto intro*: *LIMSEQ-subseq-LIMSEQ*[*OF tendsto-norm r*(*2*), *unfolded comp-def*] *simple-function-compose1*)

obtain *r'* **where** *r'*: *AE x in M*. (*λi*. (*cond-exp M F* (*λx*. *norm* (*s* (*r* (*r' i*)) *x*)) *x*)) ⟶ *cond-exp M F* (*λx*. *norm* (*f x*)) *x strict-mono r'* **using** *cond-exp-lim*[*OF integrable-norm norm-s-r*, *OF assms*] **by** *blast*

have *AE x in M*. ∀ *i*. *norm* (*cond-exp M F* (*s* (*r* (*r' i*))) *x*) ≤ *cond-exp M F* (*λx*. *norm* (*s* (*r* (*r' i*)) *x*)) *x* **using** *s* **by** (*auto intro*: *cond-exp-contraction-simple simp*

add: *AE-all-countable*)
moreover have *AE x in M*. (λi. *norm* (*cond-exp M F* (*s* (*r* (r' *i*))) *x*)) $\longrightarrow$ *norm* (*cond-exp M F f x*) **using** *r LIMSEQ-subseq-LIMSEQ*[*OF tendsto-norm* $r'(2)$, *unfolded comp-def*] **by** *fast*
ultimately show *?thesis* **using** *LIMSEQ-le* $r'(1)$ **by** *fast*
qed

lemma *cond-exp-sum* [*intro*, *simp*]:
fixes *f* :: $'t \Rightarrow 'a \Rightarrow 'b$ :: {*second-countable-topology*,*banach*}
assumes [*measurable*]: $\bigwedge i$. *integrable M* (*f i*)
shows *AE x in M*. *cond-exp M F* (λx. $\sum i{\in}I$. *f i x*) *x* = ($\sum i{\in}I$. *cond-exp M F* (*f i*) *x*)
proof (*rule has-cond-exp-charact*, *intro has-cond-expI'*)
fix *A* **assume** [*measurable*]: $A \in$ *sets F*
then have *A-meas* [*measurable*]: $A \in$ *sets M* **by** (*meson subsetD subalg subalgebra-def*)

have ($\int x{\in}A$. ($\sum i{\in}I$. *f i x*)∂M) = ($\int x$. ($\sum i{\in}I$. *indicator A x* $*_R$ *f i x*)∂M) **unfolding** *set-lebesgue-integral-def* **by** (*simp add*: *scaleR-sum-right*)
also have ... = ($\sum i{\in}I$. ($\int x$. *indicator A x* $*_R$ *f i x* ∂M)) **using** *assms* **by** (*auto intro*!: *Bochner-Integration.integral-sum integrable-mult-indicator*)
also have ... = ($\sum i{\in}I$. ($\int x$. *indicator A x* $*_R$ *cond-exp M F* (*f i*) *x* ∂M)) **using** *cond-exp-set-integral*[*OF assms*] **by** (*simp add*: *set-lebesgue-integral-def*)
also have ... = ($\int x$. ($\sum i{\in}I$. *indicator A x* $*_R$ *cond-exp M F* (*f i*) *x*)∂M) **using** *assms* **by** (*auto intro*!: *Bochner-Integration.integral-sum*[*symmetric*] *integrable-mult-indicator*)
also have ... = ($\int x{\in}A$. ($\sum i{\in}I$. *cond-exp M F* (*f i*) *x*)∂M) **unfolding** *set-lebesgue-integral-def* **by** (*simp add*: *scaleR-sum-right*)
finally show ($\int x{\in}A$. ($\sum i{\in}I$. *f i x*)∂M) = ($\int x{\in}A$. ($\sum i{\in}I$. *cond-exp M F* (*f i*) *x*)∂M) **by** *auto*
qed (*auto simp add*: *assms integrable-cond-exp*)

1.5 Ordered Real Vectors

lemma *cond-exp-gr-c*:
fixes *f* :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable M f* *AE x in M*. *f x* > *c*
shows *AE x in M*. *cond-exp M F f x* > *c*
proof −
define *X* **where** *X* = {$x \in$ *space M*. *cond-exp M F f x* $\leq$ *c*}
have [*measurable*]: $X \in$ *sets F* **unfolding** *X-def* **by** *measurable* (*metis sets.top subalg subalgebra-def*)
hence *X-in-M*: $X \in$ *sets M* **using** *sets-restr-to-subalg subalg subalgebra-def* **by** *blast*
have *emeasure M X* = *0*
proof (*rule ccontr*)
assume *emeasure M X* $\neq$ *0*
have *emeasure* (*restr-to-subalg M F*) *X* = *emeasure M X* **by** (*simp add*: *emea-*

sure-restr-to-subalg subalg)
hence *emeasure* (*restr-to-subalg M F*) *X* > *0* **using** ‹¬(*emeasure M X*) = *0*› *gr-zeroI* **by** *auto*
then obtain *A* **where** *A*: *A* ∈ *sets* (*restr-to-subalg M F*) *A* ⊆ *X* *emeasure* (*restr-to-subalg M F*) *A* > *0* *emeasure* (*restr-to-subalg M F*) *A* < ∞
using *sigma-fin-subalg* **by** (*metis emeasure-notin-sets ennreal-0 infinity-ennreal-def le-less-linear neq-top-trans not-gr-zero order-refl sigma-finite-measure.approx-PInf-emeasure-with-finite*)
hence [*simp*]: *A* ∈ *sets F* **using** *subalg sets-restr-to-subalg* **by** *blast*
hence [*simp*]: *A* ∈ *sets M* **using** *sets-restr-to-subalg subalg subalgebra-def* **by** *blast*
have [*simp*]: *set-integrable M A* (λ*x*. *c*) **using** *A subalg* **by** (*auto simp add*: *set-integrable-def emeasure-restr-to-subalg*)
have [*simp*]: *set-integrable M A f* **unfolding** *set-integrable-def* **by** (*rule integrable-mult-indicator*, *auto simp add*: *assms*(*1*))
have *AE x in M*. *indicator A x* $*_R$ *c* = *indicator A x* $*_R$ *f x*
proof (*rule integral-eq-mono-AE-eq-AE*)
show *LINT x*|*M*. *indicator A x* $*_R$ *c* = *LINT x*|*M*. *indicator A x* $*_R$ *f x*
proof (*simp only*: *set-lebesgue-integral-def*[*symmetric*], *rule antisym*)
show (∫ *x*∈*A*. *c* ∂*M*) ≤ (∫ *x*∈*A*. *f x* ∂*M*) **using** *assms*(*2*) **by** (*intro set-integral-mono-AE-banach*) *auto*
have (∫ *x*∈*A*. *f x* ∂*M*) = (∫ *x*∈*A*. *cond-exp M F f x* ∂*M*) **by** (*rule cond-exp-set-integral*, *auto simp add*: ‹*integrable M f*›)
also have ... ≤ (∫ *x*∈*A*. *c* ∂*M*) **using** *A* **by** (*auto intro*!: *set-integral-mono-banach simp add*: *X-def*)
finally show (∫ *x*∈*A*. *f x* ∂*M*) ≤ (∫ *x*∈*A*. *c* ∂*M*) **by** *simp*
qed
then have *measure M A* $*_R$ *c* = *LINT x*|*M*. *indicator A x* $*_R$ *f x* **using** *A* **by** (*auto simp*: *set-lebesgue-integral-def emeasure-restr-to-subalg subalg*)
show *AE x in M*. *indicator A x* $*_R$ *c* ≤ *indicator A x* $*_R$ *f x* **using** *assms* **by** (*auto simp add*: *X-def indicator-def*)
qed (*auto simp add*: *set-integrable-def*[*symmetric*])
then have *AE x*∈*A in M*. *c* = *f x* **by** *auto*
then have *AE x*∈*A in M*. *False* **using** *assms*(*2*) **by** *auto*
have *A* ∈ *null-sets M* **unfolding** *ae-filter-def* **by** (*meson AE-iff-null-sets* ‹*A* ∈ *sets M*› ‹*AE x*∈*A in M*. *False*›)
then show *False* **using** *A*(*3*)**by** (*simp add*: *emeasure-restr-to-subalg null-setsD1 subalg*)
qed
then show *?thesis* **using** *AE-iff-null-sets*[*OF X-in-M*] **unfolding** *X-def* **by** *auto*
qed

lemma *cond-exp-less-c*:
fixes *f* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable M f AE x in M*. *f x* < *c*
shows *AE x in M*. *cond-exp M F f x* < *c*
proof −
have *AE x in M*. *cond-exp M F f x* = − *cond-exp M F* (λ*x*. − *f x*) *x* **using** *cond-exp-uminus*[*OF assms*(*1*)] **by** *auto*

moreover have *AE x in M. cond-exp M F* ($\lambda x.\ -f\ x$) $x > -c$ **using** *assms* **by** (*intro cond-exp-gr-c*) *auto*
ultimately show *?thesis* **by** (*force simp add*: *minus-less-iff*)
qed

lemma *cond-exp-mono-strict*:
fixes $f :: {'a} \Rightarrow {'b}$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable M f integrable M g AE x in M. f x* $<$ *g x*
shows *AE x in M. cond-exp M F f x* $<$ *cond-exp M F g x*
using *cond-exp-less-c*[*OF Bochner-Integration.integrable-diff*, *OF assms*(*1*,*2*), *of 0*]
cond-exp-diff[*OF assms*(*1*,*2*)] *assms*(*3*) **by** *auto*

lemma *cond-exp-ge-c*:
fixes $f :: {'a} \Rightarrow {'b}$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes [*measurable*]: *integrable M f*
and *AE x in M. f x* $\geq$ *c*
shows *AE x in M. cond-exp M F f x* $\geq$ *c*
proof −
let *?F* = *restr-to-subalg M F*
interpret *sigma-finite-measure restr-to-subalg M F* **using** *sigma-fin-subalg* **by** *auto*
{
fix *A* **assume** *asm*: *A* ∈ *sets ?F 0* $<$ *measure ?F A*
have [*simp*]: *sets ?F* = *sets F measure ?F A* = *measure M A* **using** *asm* **by** (*auto simp add*: *measure-def sets-restr-to-subalg*[*OF subalg*] *emeasure-restr-to-subalg*[*OF subalg*])
have *M-A*: *emeasure M A* $< \infty$ **using** *measure-zero-top asm* **by** (*force simp add*: *top.not-eq-extremum*)
hence *F-A*: *emeasure ?F A* $< \infty$ **using** *asm*(*1*) *emeasure-restr-to-subalg subalg* **by** *fastforce*
have *set-lebesgue-integral M A* (λ-. *c*) ≤ *set-lebesgue-integral M A f* **using** *assms asm M-A subalg* **by** (*intro set-integral-mono-AE-banach*, *auto simp add*: *set-integrable-def integrable-mult-indicator subalgebra-def sets-restr-to-subalg*)
also have ... = *set-lebesgue-integral M A* (*cond-exp M F f*) **using** *cond-exp-set-integral*[*OF assms*(*1*)] *asm* **by** *auto*
also have ... = *set-lebesgue-integral ?F A* (*cond-exp M F f*) **unfolding** *set-lebesgue-integral-def* **using** *asm borel-measurable-cond-exp* **by** (*intro integral-subalgebra2*[*OF subalg*, *symmetric*], *simp*)
finally have (*1* / *measure ?F A*) $*_R$ *set-lebesgue-integral ?F A* (*cond-exp M F f*) ∈ {*c*..} **using** *asm subalg M-A* **by** (*auto simp add*: *set-integral-const subalgebra-def intro*!: *pos-divideR-le-eq*[*THEN iffD1*])
}
thus *?thesis* **using** *AE-restr-to-subalg*[*OF subalg*] *averaging-theorem*[*OF integrable-in-subalg closed-atLeast*, *OF subalg borel-measurable-cond-exp integrable-cond-exp*] **by** *auto*
qed

lemma *cond-exp-le-c*:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable* M f
and *AE* x *in* M. f $x \leq c$
shows *AE* x *in* M. *cond-exp* M F f $x \leq c$
proof −
have *AE* x *in* M. *cond-exp* M F f x = − *cond-exp* M F (λx. − f x) x **using** *cond-exp-uminus*[*OF assms*(*1*)] **by** *force*
moreover have *AE* x *in* M. *cond-exp* M F (λx. − f x) $x \geq$ − c **using** *assms* **by** (*intro cond-exp-ge-c*) *auto*
ultimately show *?thesis* **by** (*force simp add*: *minus-le-iff*)
qed

lemma *cond-exp-mono*:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable* M f *integrable* M g *AE* x *in* M. f $x \leq g$ x
shows *AE* x *in* M. *cond-exp* M F f $x \leq$ *cond-exp* M F g x
using *cond-exp-le-c*[*OF Bochner-Integration.integrable-diff*, *OF assms*(*1*,*2*), *of 0*]
cond-exp-diff[*OF assms*(*1*,*2*)] *assms*(*3*) **by** *auto*

lemma *cond-exp-min*:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable* M f *integrable* M g
shows *AE* ξ *in* M. *cond-exp* M F (λx. *min* (f x) (g x)) $\xi \leq$ *min* (*cond-exp* M F f ξ) (*cond-exp* M F g ξ)
proof −
have *AE* ξ *in* M. *cond-exp* M F (λx. *min* (f x) (g x)) $\xi \leq$ *cond-exp* M F f ξ **by** (*intro cond-exp-mono integrable-min assms*, *simp*)
moreover have *AE* ξ *in* M. *cond-exp* M F (λx. *min* (f x) (g x)) $\xi \leq$ *cond-exp* M F g ξ **by** (*intro cond-exp-mono integrable-min assms*, *simp*)
ultimately show *AE* ξ *in* M. *cond-exp* M F (λx. *min* (f x) (g x)) $\xi \leq$ *min* (*cond-exp* M F f ξ) (*cond-exp* M F g ξ) **by** *fastforce*
qed

lemma *cond-exp-max*:
fixes f :: $'a \Rightarrow 'b$ :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*}
assumes *integrable* M f *integrable* M g
shows *AE* ξ *in* M. *cond-exp* M F (λx. *max* (f x) (g x)) $\xi \geq$ *max* (*cond-exp* M F f ξ) (*cond-exp* M F g ξ)
proof −
have *AE* ξ *in* M. *cond-exp* M F (λx. *max* (f x) (g x)) $\xi \geq$ *cond-exp* M F f ξ **by** (*intro cond-exp-mono integrable-max assms*, *simp*)
moreover have *AE* ξ *in* M. *cond-exp* M F (λx. *max* (f x) (g x)) $\xi \geq$ *cond-exp*

M F g ξ **by** (*intro cond-exp-mono integrable-max assms*, *simp*)
ultimately show *AE ξ in M*. *cond-exp M F* (*λx*. *max* (*f x*) (*g x*)) *ξ* ≥ *max* (*cond-exp M F f ξ*) (*cond-exp M F g ξ*) **by** *fastforce*
qed

lemma *cond-exp-inf*:
fixes *f* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*, *lattice*}
assumes *integrable M f integrable M g*
shows *AE ξ in M*. *cond-exp M F* (*λx*. *inf* (*f x*) (*g x*)) *ξ* ≤ *inf* (*cond-exp M F f ξ*) (*cond-exp M F g ξ*)
unfolding *inf-min* **using** *assms* **by** (*rule cond-exp-min*)

lemma *cond-exp-sup*:
fixes *f* :: *'a* ⇒ *'b* :: {*second-countable-topology*, *banach*, *linorder-topology*, *ordered-real-vector*, *lattice*}
assumes *integrable M f integrable M g*
shows *AE ξ in M*. *cond-exp M F* (*λx*. *sup* (*f x*) (*g x*)) *ξ* ≥ *sup* (*cond-exp M F f ξ*) (*cond-exp M F g ξ*)
unfolding *sup-max* **using** *assms* **by** (*rule cond-exp-max*)

end

end
theory *Stochastic-Process*
imports *Filtration*
begin

1.6 Stochastic Process

locale *stochastic-process* = *sigma-finite-measure M* **for** *M* +
fixes *X* :: *'t* :: {*second-countable-topology*,*linorder-topology*} ⇒ *'a* ⇒ *'b*::{*real-normed-vector*, *second-countable-topology*}
assumes *random-variable*[*measurable*]: ⋀*i*. *X i* ∈ *borel-measurable M*
begin

definition *left-continuous* **where** *left-continuous* = (*AE ξ in M*. ∀ *i*. *continuous* (*at-left i*) (*λi*. *X i ξ*))
definition *right-continuous* **where** *right-continuous* = (*AE ξ in M*. ∀ *i*. *continuous* (*at-right i*) (*λi*. *X i ξ*))

lemma *compose*:
assumes ⋀*i*. *f i* ∈ *borel-measurable borel*
shows *stochastic-process M* (*λi ξ*. (*f i*) (*X i ξ*))
by (*unfold-locales*, *intro measurable-compose*[*OF random-variable assms*])

lemma *norm*: *stochastic-process M* (*λi ξ*. *norm* (*X i ξ*)) **by** (*auto intro*: *compose borel-measurable-norm*)

lemma *scaleR*:
assumes *stochastic-process M R*
shows *stochastic-process M* $(\lambda i\ \xi.\ (R\ i\ \xi) *_R (X\ i\ \xi))$
by (*unfold-locales*) (*simp add*: *borel-measurable-scaleR random-variable assms stochastic-process.random-variable*)

lemma *scaleR-const-fun*:
assumes $f \in$ *borel-measurable M*
shows *stochastic-process M* $(\lambda i\ \xi.\ f\ \xi *_R (X\ i\ \xi))$
by (*unfold-locales*, *intro borel-measurable-scaleR assms random-variable*)

lemma *scaleR-const*: *stochastic-process M* $(\lambda i\ \xi.\ c *_R (X\ i\ \xi))$ **by** (*auto intro*: *scaleR-const-fun borel-measurable-const*)

lemma *add*:
assumes *stochastic-process M Y*
shows *stochastic-process M* $(\lambda i\ \xi.\ X\ i\ \xi + Y\ i\ \xi)$
by (*unfold-locales*) (*simp add*: *borel-measurable-add random-variable assms stochastic-process.random-variable*)

lemma *diff*:
assumes *stochastic-process M Y*
shows *stochastic-process M* $(\lambda i\ \xi.\ X\ i\ \xi - Y\ i\ \xi)$
by (*unfold-locales*) (*simp add*: *borel-measurable-diff random-variable assms stochastic-process.random-variable*)

lemma *uminus*: *stochastic-process M* $(-X)$ **using** *scaleR-const*[*of* -1] **by** (*simp add*: *fun-Compl-def*)

end

1.7 Adapted Process

locale *adapted-process* = *filtered-sigma-finite-measure M F* + *stochastic-process M X* **for** *M* **and** *F* :: *'t* :: {*second-countable-topology*, *linorder-topology*, *order-bot*} ⇒ - **and** *X* :: *'t* ⇒ - ⇒ - :: {*second-countable-topology*, *banach*} +
assumes *adapted*[*measurable*]: $\bigwedge i.\ X\ i \in$ *borel-measurable* (*F i*)
begin

lemma *const-fun*:
assumes $f \in$ *borel-measurable* (*F bot*)
shows *adapted-process M F* (λ-. *f*)
using *assms* **by** (*unfold-locales*) (*blast intro*: *measurable-from-subalg subalgebra*, *metis borel-measurable-subalgebra bot.extremum sets-F-mono space-F*)

lemma *compose*:
assumes $\bigwedge i.\ f\ i \in$ *borel-measurable borel*
shows *adapted-process M F* $(\lambda i\ \xi.\ (f\ i)\ (X\ i\ \xi))$
by (*unfold-locales*, *intro measurable-compose*[*OF random-variable assms*], *intro*

measurable-compose[*OF adapted assms*])

lemma *norm*: *adapted-process M F* ($\lambda i\ \xi.$ *norm* ($X\ i\ \xi$)) **by** (*auto intro*: *compose borel-measurable-norm*)

lemma *scaleR*:
  assumes *adapted-process M F R*
  shows *adapted-process M F* ($\lambda i\ \xi.$ ($R\ i\ \xi$) $*_R$ ($X\ i\ \xi$))
proof −
  interpret *R*: *adapted-process M F R* **by** (*rule assms*)
  show *?thesis* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-scaleR adapted random-variable assms R.random-variable R.adapted*)
qed

lemma *scaleR-const-fun*:
  assumes $f \in$ *borel-measurable* (*F bot*)
  shows *adapted-process M F* ($\lambda i\ \xi.$ $f\ \xi$ $*_R$ ($X\ i\ \xi$))
  using *assms* **by** (*fast intro*: *scaleR const-fun*)

lemma *scaleR-const*: *adapted-process M F* ($\lambda i\ \xi.$ c $*_R$ ($X\ i\ \xi$)) **by** (*auto intro*: *scaleR-const-fun borel-measurable-const*)

lemma *add*:
  assumes *adapted-process M F Y*
  shows *adapted-process M F* ($\lambda i\ \xi.$ $X\ i\ \xi + Y\ i\ \xi$)
proof −
  interpret *Y*: *adapted-process M F Y* **by** (*rule assms*)
  show *?thesis* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-add adapted random-variable Y.random-variable Y.adapted*)
qed

lemma *diff*:
  assumes *adapted-process M F Y*
  shows *adapted-process M F* ($\lambda i\ \xi.$ $X\ i\ \xi - Y\ i\ \xi$)
proof −
  interpret *Y*: *adapted-process M F Y* **by** (*rule assms*)
  show *?thesis* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-diff adapted random-variable Y.random-variable Y.adapted*)
qed

lemma *uminus*: *adapted-process M F* ($-X$) **using** *scaleR-const*[*of* -1] **by** (*simp add*: *fun-Compl-def*)

end

locale *adapted-process-order* = *adapted-process M F X* **for** *M F* **and** *X* :: *'t* :: {*second-countable-topology*, *linorder-topology*, *order-bot*} ⇒ - ⇒ - :: {*linorder-topology*, *ordered-real-vector*}

1.8 Discrete-Time Processes

locale *discrete-time-stochastic-process* = *stochastic-process M X* **for** *M* **and** *X* :: *nat* ⇒ - ⇒ -
locale *discrete-time-adapted-process* = *adapted-process M F X* **for** *M F* **and** *X* :: *nat* ⇒ - ⇒ -
locale *discrete-time-adapted-process-order* = *adapted-process-order M F X* **for** *M F* **and** *X* :: *nat* ⇒ - ⇒ -

sublocale *discrete-time-adapted-process-order* ⊆ *discrete-time-adapted-process* **by** (*unfold-locales*)
sublocale *discrete-time-adapted-process* ⊆ *discrete-time-stochastic-process* **by** (*unfold-locales*)
sublocale *discrete-time-adapted-process* ⊆ *nat-filtered-sigma-finite-measure* **by** (*unfold-locales*)

1.9 Predictable Processes

context *filtered-sigma-finite-measure*
begin

definition *predictable-sigma* :: ($'t \times 'a$) *measure* **where**
predictable-sigma = *sigma* (*UNIV* × *space M*) ({{*s*<..*t*} × *A* | *A s t*. *A* ∈ *F s* ∧ *s* < *t*} ∪ {{*bot*} × *A* | *A*. *A* ∈ *F bot*})

lemma *space-predictable-sigma*[*simp*]: *space predictable-sigma* = (*UNIV* × *space M*) **unfolding** *predictable-sigma-def space-measure-of-conv* **by** *blast*

lemma *sets-predictable-sigma*[*simp*]: *sets predictable-sigma* = *sigma-sets* (*UNIV* × *space M*) ({{*s*<..*t*} × *A* | *A s t*. *A* ∈ *F s* ∧ *s* < *t*} ∪ {{*bot*} × *A* | *A*. *A* ∈ *F bot*})
unfolding *predictable-sigma-def sets-measure-of-conv*
using *space-F sets.sets-into-space*
by (*fastforce intro*!: *if-P*)

definition *predictable* :: ($'t \Rightarrow 'a \Rightarrow 'b$ :: {*second-countable-topology*,*banach*}) ⇒ *bool* **where**
predictable X = (*case-prod X* ∈ *borel-measurable* (*predictable-sigma*))

lemmas *predictableD* = *measurable-sets*[*OF predictable-def*[*THEN iffD1*], *unfolded space-predictable-sigma*]

lemma (**in** *nat-filtered-sigma-finite-measure*) *predictable-sets-in-F*:
assumes ($\bigcup$*i*. {*i*} × *A i*) ∈ *predictable-sigma*
shows *A* (*Suc i*) ∈ *F i*
A 0 ∈ *F 0*
using *assms* **unfolding** *sets-predictable-sigma*
proof (*induction* ($\bigcup$*i*. {*i*} × *A i*) *arbitrary*: *A*)
case *Basic*
{
assume ∃*S*. ($\bigcup$*i*. {*i*} × *A i*) = {*bot*} × *S*
then obtain *S* **where** *S*: ($\bigcup$*i*. {*i*} × *A i*) = {*bot*} × *S* **by** *blast*
hence *S* ∈ *F 0* **using** *Basic* **by** (*fastforce simp add*: *times-eq-iff bot-nat-def*)

moreover have *A i* = {} **if** *i* ≠ *bot* **for** *i* **using** *that S* **by** *blast*
moreover have *A bot* = *S* **using** *S* **by** *blast*
ultimately have *A* (*Suc i*) ∈ *F i A 0* ∈ *F 0* **for** *i* **unfolding** *bot-nat-def* **by** (*auto simp add*: *bot-nat-def*)
}
note ∗ = *this*
{
assume ∄*S*. (⋃*i*. {*i*} × *A i*) = {*bot*} × *S*
then obtain *s t B* **where** *B*: (⋃*i*. {*i*} × *A i*) = {*s*<..*t*} × *B B* ∈ *sets* (*F s*) *s* < *t* **using** *Basic* **by** *auto*
hence *A i* = *B* **if** *i* ∈ {*s*<..*t*} **for** *i* **using** *that* **by** *fast*
moreover have *A i* = {} **if** *i* ∉ {*s*<..*t*} **for** *i* **using** *B that* **by** *fastforce*
ultimately have *A* (*Suc i*) ∈ *F i A 0* ∈ *F 0* **for** *i* **unfolding** *bot-nat-def* **using** *B sets-F-mono* **by** (*auto simp add*: *bot-nat-def*) (*metis less-Suc-eq-le sets.empty-sets subset-eq*)
}
note ∗∗ = *this*
show *A* (*Suc i*) ∈ *sets* (*F i*) *A 0* ∈ *sets* (*F 0*) **using** ∗(*1*)[*of i*] ∗(*2*) ∗∗(*1*)[*of i*] ∗∗(*2*) **by** *auto blast*+
next
case *Empty*
{
case *1*
then show *?case* **using** *Empty* **by** *simp*
next
case *2*
then show *?case* **using** *Empty* **by** *simp*
}
next
case (*Compl a*)
have *a-in*: *a* ⊆ *UNIV* × *space M* **using** *Compl*(*1*) *sets.sets-into-space sets-predictable-sigma space-predictable-sigma* **by** *metis*
hence *A-in*: *A i* ⊆ *space M* **for** *i* **using** *Compl*(*4*) **by** *blast*
have *a*: *a* = *UNIV* × *space M* − (⋃*i*. {*i*} × *A i*) **using** *a-in Compl*(*4*) **by** *blast*
also have ... = (⋃*j*. {*j*} × (*space M* − *A j*)) **by** *blast*
finally have ∗: (*space M* − *A* (*Suc i*)) ∈ *F i* (*space M* − *A 0*) ∈ *F 0* **using** *Compl*(*2*,*3*) **by** *auto*
{
case *1*
then show *?case* **using** ∗ *A-in* **by** (*metis double-diff sets.compl-sets space-F subset-refl*)
next
case *2*
then show *?case* **using** ∗ *A-in* **by** (*metis double-diff sets.compl-sets space-F subset-refl*)
}
next
case (*Union a*)
have *a-in*: *a i* ⊆ *UNIV* × *space M* **for** *i* **using** *Union*(*1*) *sets.sets-into-space*

sets-predictable-sigma space-predictable-sigma **by** *metis*
hence *A-in*: $A\ i \subseteq space\ M$ **for** *i* **using** *Union(4)* **by** *blast*
have $snd\ x \in snd\ `\ (a\ i \cap (\{fst\ x\} \times space\ M))$ **if** $x \in a\ i$ **for** *i x* **using** *that a-in* **by** *fastforce*
hence *a-i*: $a\ i = (\bigcup j.\ \{j\} \times (snd\ `\ (a\ i \cap (\{j\} \times space\ M))))$ **for** *i* **by** *force*
have *A-i*: $A\ i = snd\ `\ (\bigcup\ (range\ a) \cap (\{i\} \times space\ M))$ **for** *i* **unfolding** *Union(4)* **using** *A-in* **by** *force*
have $*$: $snd\ `\ (a\ j \cap (\{Suc\ i\} \times space\ M)) \in F\ i\ snd\ `\ (a\ j \cap (\{0\} \times space\ M)) \in F\ 0$ **for** *j* **using** *Union(2,3)[OF a-i]* **by** *auto*
{
case *1*
have $(\bigcup j.\ snd\ `\ (a\ j \cap (\{Suc\ i\} \times space\ M))) \in F\ i$ **using** $*$ **by** *fast*
moreover have $(\bigcup j.\ snd\ `\ (a\ j \cap (\{Suc\ i\} \times space\ M))) = snd\ `\ (\bigcup\ (range\ a) \cap (\{Suc\ i\} \times space\ M))$ **by** *fast*
ultimately show *?case* **using** *A-i* **by** *metis*
next
case *2*
have $(\bigcup j.\ snd\ `\ (a\ j \cap (\{0\} \times space\ M))) \in F\ 0$ **using** $*$ **by** *fast*
moreover have $(\bigcup j.\ snd\ `\ (a\ j \cap (\{0\} \times space\ M))) = snd\ `\ (\bigcup\ (range\ a) \cap (\{0\} \times space\ M))$ **by** *fast*
ultimately show *?case* **using** *A-i* **by** *metis*
}
qed

lemma (**in** *nat-filtered-sigma-finite-measure*) *predictable-discrete-time-process-measurable*:
assumes *predictable X*
shows $X\ i \in borel\text{-}measurable\ (F\ (i - 1))$
proof (*cases i*)
case *0*
{
fix $S :: 'b\ set$ **assume** *open-S*: *open S*
hence $\{0\} \times space\ M \in predictable\text{-}sigma$ **by** (*auto simp add*: *bot-nat-def space-F[symmetric, of bot]*)
moreover have $case\text{-}prod\ X\ -`\ S \cap (UNIV \times space\ M) \in predictable\text{-}sigma$ **using** *open-S* **by** (*intro predictableD[OF assms]*, *simp add*: *borel-open*)
ultimately have $case\text{-}prod\ X\ -`\ S \cap (\{0\} \times space\ M) \in predictable\text{-}sigma$ **unfolding** *sets-predictable-sigma* **using** *space-F sets.sets-into-space*
by (*subst Times-Int-distrib1[of {0} UNIV space M, simplified]*, *subst inf.commute[of - × -]*, *subst Int-assoc[symmetric]*, *subst Int-range-binary*)
(*intro sigma-sets-Inter binary-in-sigma-sets*, *fast*)+
moreover have $case\text{-}prod\ X\ -`\ S \cap (\{0\} \times space\ M) = \{0\} \times (X\ 0\ -`\ S \cap space\ M)$ **by** (*auto simp add*: *le-Suc-eq*)
moreover have $... = (\bigcup i.\ \{i\} \times (if\ i = 0\ then\ X\ 0\ -`\ S \cap space\ M\ else\ \{\}))$ **by** (*auto split*: *if-splits*)
ultimately have $(\bigcup i.\ \{i\} \times (if\ i = 0\ then\ X\ 0\ -`\ S \cap space\ M\ else\ \{\})) \in predictable\text{-}sigma$ **by** *argo*
then have $X\ 0\ -`\ S \cap space\ M \in sets\ (F\ 0)$ **using** *predictable-sets-in-F*[*of* $\lambda i.\ if\ i = 0\ then\ X\ 0\ -`\ S \cap space\ M\ else\ \{\}$] **by** *presburger*
}

hence *X 0* ∈ *borel-measurable* (*F 0*) **by** (*fastforce simp add*: *bot-nat-def space-F intro*!: *borel-measurableI*)
thus *?thesis* **using** *0* **by** *force*
next
case (*Suc i*)
{
fix *S* :: *'b set* **assume** *open-S*: *open S*
have {*Suc i*} = {*i*<..*Suc i*} **by** *fastforce*
hence {*Suc i*} × *space M* ∈ *predictable-sigma* **unfolding** *space-F*[*symmetric*, *of i*] **by** (*auto intro*!: *sigma-sets.Basic*)
moreover have *case-prod X* −' *S* ∩ (*UNIV* × *space M*) ∈ *predictable-sigma* **using** *open-S* **by** (*intro predictableD*[*OF assms*], *simp add*: *borel-open*)
ultimately have *case-prod X* −' *S* ∩ ({*Suc i*} × *space M*) ∈ *predictable-sigma* **unfolding** *sets-predictable-sigma* **using** *space-F sets.sets-into-space*
by (*subst Times-Int-distrib1*[*of* {*Suc i*} *UNIV space M*, *simplified*], *subst inf.commute*[*of* - × -], *subst Int-assoc*[*symmetric*], *subst Int-range-binary*)
(*intro sigma-sets-Inter binary-in-sigma-sets*, *fast*)+
moreover have *case-prod X* −' *S* ∩ ({*Suc i*} × *space M*) = {*Suc i*} × (*X* (*Suc i*) −' *S* ∩ *space M*) **by** (*auto simp add*: *le-Suc-eq*)
moreover have ... = (⋃*j*. {*j*} × (*if j* = *Suc i then* (*X* (*Suc i*) −' *S* ∩ *space M*) *else* {})) **by** (*auto split*: *if-splits*)
ultimately have (⋃*j*. {*j*} × (*if j* = *Suc i then* (*X* (*Suc i*) −' *S* ∩ *space M*) *else* {})) ∈ *predictable-sigma* **by** *argo*
then have *X* (*Suc i*) −' *S* ∩ *space M* ∈ *sets* (*F i*) **using** *predictable-sets-in-F*[*of* λ*j*. *if j* = *Suc i then* (*X* (*Suc i*) −' *S* ∩ *space M*) *else* {}] **by** *presburger*
}
hence *X* (*Suc i*) ∈ *borel-measurable* (*F i*) **by** (*fastforce simp add*: *space-F intro*!: *borel-measurableI*)
then show *?thesis* **using** *Suc* **by** *force*
qed

end

end
theory *Martingale*
imports *Stochastic-Process Conditional-Expectation-Banach*
begin

1.10 Martingale

unbundle *lattice-syntax*

locale *martingale* = *adapted-process* +
assumes *integrable*: ⋀*i*. *integrable M* (*X i*)
and *martingale-property*: ⋀*i j*. *i* ≤ *j* ⟹ *AE* ξ *in M*. *X i* ξ = *cond-exp M* (*F i*) (*X j*) ξ

lemma (**in** *filtered-sigma-finite-measure*) *martingale-const*[*intro*]:
assumes *integrable M f f* ∈ *borel-measurable* (*F* ⊥)

shows *martingale M F* (λ-. *f*)
using *assms cond-exp-F-meas*[*OF assms*(*1*), *THEN AE-symmetric*]
by (*unfold-locales*)
(*simp add*: *borel-measurable-integrable*,
metis bot.extremum measurable-from-subalg sets-F-mono space-F subalgebra-def, *blast*,
metis (*mono-tags*, *lifting*) *borel-measurable-subalgebra bot-least filtration.sets-F-mono filtration-axioms space-F*)

lemma (**in** *filtered-sigma-finite-measure*) *martingale-cond-exp*[*intro*]:
assumes *integrable M f*
shows *martingale M F* (λi. *cond-exp M* (*F i*) *f*)
by (*unfold-locales*,
auto simp add: *subalgebra borel-measurable-cond-exp borel-measurable-cond-exp′*
intro!: *cond-exp-nested-subalg*[*OF assms*],
simp add: *sets-F-mono space-F subalgebra-def*)

1.11 Submartingale

locale *submartingale* = *adapted-process-order* +
assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
and *submartingale-property*: $\bigwedge i\ j.\ i \leq j \Longrightarrow$ *AE* ξ *in M*. *X i* $\xi \leq$ *cond-exp M* (*F i*) (*X j*) ξ

1.12 Supermartingale

locale *supermartingale* = *adapted-process-order* +
assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
and *supermartingale-property*: $\bigwedge i\ j.\ i \leq j \Longrightarrow$ *AE* ξ *in M*. *X i* $\xi \geq$ *cond-exp M* (*F i*) (*X j*) ξ

1.13 Martingale Stuff

locale *martingale-order* = *martingale M F X* **for** *M F* **and** *X* :: - $\Rightarrow$ - $\Rightarrow$ - :: {*linorder-topology*, *ordered-real-vector*}
begin

lemma *is-submartingale*: *submartingale M F X* **using** *martingale-property* **by** (*unfold-locales*) (*force simp add*: *integrable*)+

lemma *is-supermartingale*: *supermartingale M F X* **using** *martingale-property* **by** (*unfold-locales*) (*force simp add*: *integrable*)+

end

sublocale *martingale-order* $\subseteq$ *martingale-is-submartingale*: *submartingale* **by** (*rule is-submartingale*)

sublocale *martingale-order* $\subseteq$ *martingale-is-supermartingale*: *supermartingale* **by** (*rule is-supermartingale*)

locale *submartingale-lattice* = *submartingale M F X* **for** *M F* **and** *X* :: - ⇒ - ⇒ - :: {*linorder-topology*, *lattice*, *ordered-real-vector*}

locale *supermartingale-lattice* = *supermartingale M F X* **for** *M F* **and** *X* :: - ⇒ - ⇒ - :: {*linorder-topology*, *lattice*, *ordered-real-vector*}

locale *martingale-lattice* = *martingale M F X* **for** *M F* **and** *X* :: - ⇒ - ⇒ - :: {*linorder-topology*, *lattice*, *ordered-real-vector*}
begin

lemma *is-submartingale*: *submartingale-lattice M F X* **using** *martingale-property*
by (*unfold-locales*) (*force simp add*: *integrable*)+

lemma *is-supermartingale*: *supermartingale-lattice M F X* **using** *martingale-property*
by (*unfold-locales*) (*force simp add*: *integrable*)+

end

sublocale *martingale-lattice* ⊆ *martingale-is-submartingale*: *submartingale-lattice*
by (*rule is-submartingale*)

sublocale *martingale-lattice* ⊆ *martingale-is-supermartingale*: *supermartingale-lattice*
by (*rule is-supermartingale*)

context *martingale*
begin

lemma *set-integral-eq*:
  assumes $A \in F\ i\ i \leq j$
  shows *set-lebesgue-integral M A* (*X i*) = *set-lebesgue-integral M A* (*X j*)
proof −
  have $\int x \in A.\ X\ i\ x\ \partial M = \int x \in A.$ *cond-exp M* (*F i*) (*X j*) *x* ∂M **using** *martingale-property*[*OF assms*(*2*)] *borel-measurable-cond-exp′ assms*(*1*) *subalgebra subalgebra-def* **by** (*intro set-lebesgue-integral-cong-AE*[*OF - random-variable*]) *fast-force*+
  also have ... = $\int x \in A.\ X\ j\ x\ \partial M$ **using** *assms*(*1*) **by** (*auto simp*: *integrable intro*: *cond-exp-set-integral*[*symmetric*])
  finally show *?thesis* **.**
qed

lemma *scaleR-const*[*intro*]:
  shows *martingale M F* ($\lambda i\ x.\ c *_R X\ i\ x$)
proof −
  {
    fix *i j* :: *′b* **assume** $i \leq j$
    hence *AE x in M*. $c *_R X\ i\ x$ = *cond-exp M* (*F i*) ($\lambda x.\ c *_R X\ j\ x$) *x*
      using *cond-exp-scaleR-right*[*OF integrable*, *of i c*, *THEN AE-symmetric*] *martingale-property* **by** *force*

```
  }
  thus ?thesis by (unfold-locales) (auto simp add: borel-measurable-const-scaleR
adapted random-variable integrable)
qed

lemma uminus[intro]:
  shows martingale M F (− X)
  using scaleR-const[of −1] by (force intro: back-subst[of martingale M F])

lemma add[intro]:
  assumes martingale M F Y
  shows martingale M F (λi ξ. X i ξ + Y i ξ)
proof −
  interpret Y: martingale M F Y by (rule assms)
  {
    fix i j :: 'b assume asm: i ≤ j
    have AE ξ in M. X i ξ + Y i ξ = cond-exp M (F i) (λx. X j x + Y j x) ξ
      using cond-exp-add[OF integrable martingale.integrable[OF assms], of i j j,
THEN AE-symmetric]
            martingale-property[OF asm] martingale.martingale-property[OF assms
asm] by force
  }
  thus ?thesis using assms
  by (unfold-locales) (auto simp add: borel-measurable-add random-variable adapted
integrable Y.adapted Y.random-variable martingale.integrable)
qed

lemma diff[intro]:
  assumes martingale M F Y
  shows martingale M F (λi x. X i x − Y i x)
proof −
  interpret Y: martingale M F Y by (rule assms)
  {
    fix i j :: 'b assume asm: i ≤ j
    have AE ξ in M. X i ξ − Y i ξ = cond-exp M (F i) (λx. X j x − Y j x) ξ
      using cond-exp-diff[OF integrable martingale.integrable[OF assms], of i j j,
THEN AE-symmetric, unfolded fun-diff-def]
            martingale-property[OF asm] martingale.martingale-property[OF assms
asm] by fastforce
  }
 thus ?thesis using assms by (unfold-locales) (auto simp add: borel-measurable-diff
random-variable adapted integrable Y.random-variable Y.adapted martingale.integrable)

qed

end

lemma (in adapted-process) martingale-of-set-integral-eq:
  assumes integrable: ⋀i. integrable M (X i)
```

and $\bigwedge A\ i\ j.\ i \leq j \Longrightarrow A \in F\ i \Longrightarrow$ *set-lebesgue-integral M A* $(X\ i) =$ *set-lebesgue-integral M A* $(X\ j)$
shows *martingale M F X*
proof (*unfold-locales*)
fix *i j* :: *'t* **assume** *asm*: $i \leq j$
interpret *sigma-finite-measure restr-to-subalg M* (*F i*) **by** (*simp add*: *sigma-fin-subalg*)
{
fix *A* **assume** $A \in$ *restr-to-subalg M* (*F i*)
hence $*$: $A \in F\ i$ **using** *sets-restr-to-subalg subalgebra* **by** *blast*
have *set-lebesgue-integral* (*restr-to-subalg M* (*F i*)) *A* (*X i*) = *set-lebesgue-integral M A* (*X i*) **using** $*$ *subalg* **by** (*auto simp*: *set-lebesgue-integral-def intro*: *integral-subalgebra2 borel-measurable-scaleR adapted borel-measurable-indicator*)
also have ... = *set-lebesgue-integral M A* (*cond-exp M* (*F i*) (*X j*)) **using** $*$ *assms*(*2*)[*OF asm*] *cond-exp-set-integral*[*OF integrable*] **by** *auto*
finally have *set-lebesgue-integral* (*restr-to-subalg M* (*F i*)) *A* (*X i*) = *set-lebesgue-integral* (*restr-to-subalg M* (*F i*)) *A* (*cond-exp M* (*F i*) (*X j*)) **using** $*$ *subalg* **by** (*auto simp*: *set-lebesgue-integral-def intro*!: *integral-subalgebra2*[*symmetric*] *borel-measurable-scaleR borel-measurable-cond-exp borel-measurable-indicator*)
}
hence *AE* ξ *in restr-to-subalg M* (*F i*). *X i* ξ = *cond-exp M* (*F i*) (*X j*) ξ **by** (*intro density-unique*, *auto intro*: *integrable-in-subalg subalg borel-measurable-cond-exp integrable*)
thus *AE* ξ *in M*. *X i* ξ = *cond-exp M* (*F i*) (*X j*) ξ **using** *AE-restr-to-subalg*[*OF subalg*] **by** *blast*
qed (*simp add*: *integrable*)

lemma *martingale-orderI*:
assumes *submartingale M F X supermartingale M F X*
shows *martingale-order M F X*
proof −
interpret *submartingale M F X* **by** (*rule assms*)
interpret *supermartingale M F X* **by** (*rule assms*)
show *?thesis* **using** *integrable submartingale-property supermartingale-property* **by** (*unfold-locales*) (*fast intro*: *antisym*)+
qed

lemma *martingale-iff*: *martingale M F X* $\longleftrightarrow$ *submartingale M F X* $\wedge$ *supermartingale M F X*
using *martingale-orderI martingale-order.is-submartingale martingale-order.is-supermartingale martingale-order-def* **by** *blast*

1.14 Submartingale Stuff

context *submartingale*
begin

lemma *set-integral-le*:
assumes $A \in F\ i\ i \leq j$
shows *set-lebesgue-integral M A* (*X i*) $\leq$ *set-lebesgue-integral M A* (*X j*)

unfolding *cond-exp-set-integral*[*OF integrable assms*(*1*), *of j*]
using *submartingale-property*[*OF assms*(*2*)]
by (*simp only*: *set-lebesgue-integral-def*, *intro integral-mono-AE-banach*, *metis assms*(*1*) *in-mono integrable integrable-mult-indicator subalgebra subalgebra-def*, *metis assms*(*1*) *in-mono integrable-mult-indicator subalgebra subalgebra-def integrable-cond-exp*)
(*auto intro*: *scaleR-left-mono*)

lemma *cond-exp-diff-nonneg*:
assumes $i \leq j$
shows *AE x in M*. $0 \leq$ *cond-exp M* (*F i*) ($\lambda\xi$. *X j* ξ $-$ *X i* ξ) *x*
using *submartingale-property*[*OF assms*] *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i j i*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] **by** *fastforce*

lemma *add*[*intro*]:
assumes *submartingale M F Y*
shows *submartingale M F* ($\lambda i\ \xi$. *X i* ξ $+$ *Y i* ξ)
proof $-$
interpret *Y*: *submartingale M F Y* **by** (*rule assms*)
{
fix *i j* :: *'b* **assume** *asm*: $i \leq j$
have *AE* ξ *in M*. *X i* ξ $+$ *Y i* ξ $\leq$ *cond-exp M* (*F i*) (λx. *X j x* $+$ *Y j x*) ξ
using *cond-exp-add*[*OF integrable submartingale.integrable*[*OF assms*], *of i j j*]
submartingale-property[*OF asm*] *submartingale.submartingale-property*[*OF assms asm*] *add-mono*[*of X i - - Y i -*] **by** *force*
}
thus *?thesis* **using** *assms* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-add random-variable adapted integrable Y.random-variable Y.adapted submartingale.integrable*)

qed

lemma *diff*[*intro*]:
assumes *supermartingale M F Y*
shows *submartingale M F* ($\lambda i\ \xi$. *X i* ξ $-$ *Y i* ξ)
proof $-$
interpret *Y*: *supermartingale M F Y* **by** (*rule assms*)
{
fix *i j* :: *'b* **assume** *asm*: $i \leq j$
have *AE* ξ *in M*. *X i* ξ $-$ *Y i* ξ $\leq$ *cond-exp M* (*F i*) (λx. *X j x* $-$ *Y j x*) ξ
using *cond-exp-diff*[*OF integrable supermartingale.integrable*[*OF assms*], *of i j j*, *unfolded fun-diff-def*]
submartingale-property[*OF asm*] *supermartingale.supermartingale-property*[*OF assms asm*] *diff-mono*[*of X i - - - Y i -*] **by** *force*
}
thus *?thesis* **using** *assms* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-diff random-variable adapted integrable Y.random-variable Y.adapted supermartingale.integrable*)

qed

lemma *scaleR-nonneg*:
  assumes $c \geq 0$
  shows *submartingale M F* $(\lambda i\ \xi.\ c *_R X\ i\ \xi)$
proof
  {
    fix *i j* :: *'b* **assume** *asm*: $i \leq j$
    show *AE* ξ *in M*. $c *_R X\ i\ \xi \leq$ *cond-exp M* $(F\ i)$ $(\lambda\xi.\ c *_R X\ j\ \xi)\ \xi$
     using *cond-exp-scaleR-right*[*OF integrable*, *of i c j*] *submartingale-property*[*OF asm*] **by** (*auto intro*!: *scaleR-left-mono*[*OF - assms*])
  }
qed (*auto simp add*: *borel-measurable-integrable borel-measurable-scaleR integrable random-variable adapted borel-measurable-const-scaleR*)

lemma *scaleR-nonpos*:
  assumes $c \leq 0$
  shows *supermartingale M F* $(\lambda i\ \xi.\ c *_R X\ i\ \xi)$
proof
  {
    fix *i j* :: *'b* **assume** *asm*: $i \leq j$
    show *AE* ξ *in M*. $c *_R X\ i\ \xi \geq$ *cond-exp M* $(F\ i)$ $(\lambda\xi.\ c *_R X\ j\ \xi)\ \xi$
     using *cond-exp-scaleR-right*[*OF integrable*, *of i c j*] *submartingale-property*[*OF asm*] **by** (*auto intro*!: *scaleR-left-mono-neg*[*OF - assms*])
  }
qed (*auto simp add*: *borel-measurable-integrable borel-measurable-scaleR integrable random-variable adapted borel-measurable-const-scaleR*)

lemma *uminus*[*intro*]:
  shows *supermartingale M F* $(-\ X)$
  unfolding *fun-Compl-def* **using** *scaleR-nonpos*[*of* -1] **by** *simp*

lemma *max*:
  assumes *submartingale M F Y*
  shows *submartingale M F* $(\lambda i\ \xi.\ max\ (X\ i\ \xi)\ (Y\ i\ \xi))$
proof (*unfold-locales*)
  interpret *Y*: *submartingale M F Y* **by** (*rule assms*)
  {
    fix *i j* :: *'b* **assume** *asm*: $i \leq j$
     have *AE* ξ *in M*. *max* $(X\ i\ \xi)\ (Y\ i\ \xi) \leq$ *max* (*cond-exp M* $(F\ i)\ (X\ j)\ \xi$) (*cond-exp M* $(F\ i)\ (Y\ j)\ \xi$) **using** *submartingale-property Y.submartingale-property asm* **unfolding** *max-def* **by** *fastforce*
    thus *AE* ξ *in M*. *max* $(X\ i\ \xi)\ (Y\ i\ \xi) \leq$ *cond-exp M* $(F\ i)$ $(\lambda\xi.\ max\ (X\ j\ \xi)\ (Y\ j\ \xi))\ \xi$ **using** *cond-exp-max*[*OF integrable Y.integrable*, *of i j j*] *order.trans* **by** *fast*
  }
  show $\bigwedge i.\ (\lambda\xi.\ max\ (X\ i\ \xi)\ (Y\ i\ \xi)) \in$ *borel-measurable M* $\bigwedge i.\ (\lambda\xi.\ max\ (X\ i\ \xi)\ (Y\ i\ \xi)) \in$ *borel-measurable* $(F\ i)$ $\bigwedge i.$ *integrable M* $(\lambda\xi.\ max\ (X\ i\ \xi)\ (Y\ i\ \xi))$ **by** (*force intro*: *Y.integrable integrable assms*)+
qed

lemma *max-0*:
shows *submartingale M F* ($\lambda i\ \xi.\ max\ 0\ (X\ i\ \xi)$)
proof −
interpret *zero*: *submartingale M F* λ- -. *0* **by** (*intro martingale-order.is-submartingale*, *unfold-locales*, *auto*)
show *?thesis* **by** (*intro zero.max submartingale-axioms*)
qed

end

lemma (**in** *submartingale-lattice*) *sup*:
assumes *submartingale-lattice M F Y*
shows *submartingale-lattice M F* ($\lambda i\ \xi.\ sup\ (X\ i\ \xi)\ (Y\ i\ \xi)$)
using *submartingale-lattice.intro submartingale.max*[*OF submartingale-axioms assms*[*THEN submartingale-lattice.axioms*]] **unfolding** *sup-max*[*symmetric*] **.**

lemma (**in** *adapted-process-order*) *submartingale-of-cond-exp-diff-nonneg*:
assumes *integrable*: $\bigwedge i.$ *integrable M* ($X\ i$)
and *diff-nonneg*: $\bigwedge i\ j.\ i \le j \Longrightarrow$ *AE x in M*. $0 \le$ *cond-exp M* ($F\ i$) ($\lambda\xi.\ X\ j\ \xi - X\ i\ \xi$) x
shows *submartingale M F X*
proof (*unfold-locales*)
{
fix $i\ j$ :: $'t$ **assume** *asm*: $i \le j$
show *AE* ξ *in M*. $X\ i\ \xi \le$ *cond-exp M* ($F\ i$) ($X\ j$) ξ
using *diff-nonneg*[*OF asm*] *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i j i*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] **by** *fastforce*
}
qed (*intro integrable*)

lemma (**in** *adapted-process-order*) *submartingale-of-set-integral-le*:
assumes *integrable*: $\bigwedge i.$ *integrable M* ($X\ i$)
and $\bigwedge A\ i\ j.\ i \le j \Longrightarrow A \in F\ i \Longrightarrow$ *set-lebesgue-integral M A* ($X\ i$) $\le$ *set-lebesgue-integral M A* ($X\ j$)
shows *submartingale M F X*
proof (*unfold-locales*)
{
fix $i\ j$ :: $'t$ **assume** *asm*: $i \le j$
interpret *sigma-finite-measure restr-to-subalg M* ($F\ i$) **by** (*simp add*: *sigma-fin-subalg*)
{
fix A **assume** $A \in$ *restr-to-subalg M* ($F\ i$)
hence $*$: $A \in F\ i$ **using** *sets-restr-to-subalg subalgebra* **by** *blast*
have *set-lebesgue-integral* (*restr-to-subalg M* ($F\ i$)) A ($X\ i$) = *set-lebesgue-integral M A* ($X\ i$) **using** $*$ *subalg* **by** (*auto simp*: *set-lebesgue-integral-def intro*: *integral-subalgebra2 borel-measurable-scaleR adapted borel-measurable-indicator*)
also have ... $\le$ *set-lebesgue-integral M A* (*cond-exp M* ($F\ i$) ($X\ j$)) **using** $*$ *assms*(*2*)[*OF asm*] *cond-exp-set-integral*[*OF integrable*] **by** *auto*
also have ... = *set-lebesgue-integral* (*restr-to-subalg M* ($F\ i$)) A (*cond-exp*

M (*F i*) (*X j*)) **using** ∗ *subalg* **by** (*auto simp*: *set-lebesgue-integral-def intro*!: *integral-subalgebra2*[*symmetric*] *borel-measurable-scaleR borel-measurable-cond-exp borel-measurable-indicator*)

finally have $0 \leq$ *set-lebesgue-integral* (*restr-to-subalg M* (*F i*)) *A* ($\lambda\xi$. *cond-exp M* (*F i*) (*X j*) ξ − *X i* ξ) **using** ∗ *subalg* **by** (*subst set-integral-diff*, *auto simp add*: *set-integrable-def sets-restr-to-subalg intro*!: *integrable adapted integrable-in-subalg borel-measurable-scaleR borel-measurable-indicator borel-measurable-cond-exp integrable-mult-indicator*)

}

hence *AE* ξ *in restr-to-subalg M* (*F i*). $0 \leq$ *cond-exp M* (*F i*) (*X j*) ξ − *X i* ξ **by** (*intro density-nonneg integrable-in-subalg subalg borel-measurable-diff borel-measurable-cond-exp adapted Bochner-Integration.integrable-diff integrable-cond-exp integrable*)

thus *AE* ξ *in M*. *X i* $\xi \leq$ *cond-exp M* (*F i*) (*X j*) ξ **using** *AE-restr-to-subalg*[*OF subalg*] **by** *simp*

}

qed (*intro integrable*)

1.15 Supermartingale Stuff

context *supermartingale*
begin

lemma *set-integral-ge*:
assumes *A* ∈ *F i* $i \leq j$
shows *set-lebesgue-integral M A* (*X i*) ≥ *set-lebesgue-integral M A* (*X j*)
unfolding *cond-exp-set-integral*[*OF integrable assms*(*1*), *of j*]
using *supermartingale-property*[*OF assms*(*2*)]
by (*simp only*: *set-lebesgue-integral-def*, *intro integral-mono-AE-banach*, *metis assms*(*1*) *in-mono integrable-mult-indicator subalgebra subalgebra-def integrable-cond-exp*, *metis assms*(*1*) *in-mono integrable integrable-mult-indicator subalgebra subalgebra-def*)
(*auto intro*: *scaleR-left-mono*)

lemma *cond-exp-diff-nonneg*:
assumes $i \leq j$
shows *AE x in M*. $0 \leq$ *cond-exp M* (*F i*) ($\lambda\xi$. *X i* ξ − *X j* ξ) *x*
using *supermartingale-property*[*OF assms*] *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i i j*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] **by** *fastforce*

lemma *add*[*intro*]:
assumes *supermartingale M F Y*
shows *supermartingale M F* ($\lambda i\ \xi$. *X i* ξ + *Y i* ξ)
proof −
interpret *Y*: *supermartingale M F Y* **by** (*rule assms*)
{
fix *i j* :: *'b* **assume** *asm*: $i \leq j$
have *AE* ξ *in M*. *X i* ξ + *Y i* ξ ≥ *cond-exp M* (*F i*) (λx. *X j x* + *Y j x*) ξ
using *cond-exp-add*[*OF integrable supermartingale.integrable*[*OF assms*], *of i j j*]

supermartingale-property[*OF asm*] *supermartingale.supermartingale-property*[*OF assms asm*] *add-mono*[*of - X i - - Y i -*] **by** *force*
}
thus *?thesis* **using** *assms* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-add random-variable adapted integrable Y.random-variable Y.adapted supermartingale.integrable*)

qed

lemma *diff*[*intro*]:
assumes *submartingale M F Y*
shows *supermartingale M F* ($\lambda i\ \xi.\ X\ i\ \xi\ -\ Y\ i\ \xi$)
proof −
interpret *Y*: *submartingale M F Y* **by** (*rule assms*)
{
fix *i j* :: *'b* **assume** *asm*: $i \le j$
have *AE* ξ *in M*. $X\ i\ \xi - Y\ i\ \xi \ge$ *cond-exp M* (*F i*) ($\lambda x.\ X\ j\ x - Y\ j\ x$) ξ
using *cond-exp-diff*[*OF integrable submartingale.integrable*[*OF assms*], *of i j j*, *unfolded fun-diff-def*]
supermartingale-property[*OF asm*] *submartingale.submartingale-property*[*OF assms asm*] *diff-mono*[*of - X i - Y i -*] **by** *force*
}
thus *?thesis* **using** *assms* **by** (*unfold-locales*) (*auto simp add*: *borel-measurable-diff random-variable adapted integrable Y.random-variable Y.adapted submartingale.integrable*)

qed

lemma *scaleR-nonneg*:
assumes $c \ge 0$
shows *supermartingale M F* ($\lambda i\ \xi.\ c *_R X\ i\ \xi$)
proof
{
fix *i j* :: *'b* **assume** *asm*: $i \le j$
show *AE* ξ *in M*. $c *_R X\ i\ \xi \ge$ *cond-exp M* (*F i*) ($\lambda\xi.\ c *_R X\ j\ \xi$) ξ
using *cond-exp-scaleR-right*[*OF integrable*, *of i c j*] *supermartingale-property*[*OF asm*]
by (*auto intro*!: *scaleR-left-mono*[*OF - assms*])
}
qed (*auto simp add*: *borel-measurable-integrable borel-measurable-scaleR integrable random-variable adapted borel-measurable-const-scaleR*)

lemma *scaleR-nonpos*:
assumes $c \le 0$
shows *submartingale M F* ($\lambda i\ \xi.\ c *_R X\ i\ \xi$)
proof
{
fix *i j* :: *'b* **assume** *asm*: $i \le j$
show *AE* ξ *in M*. $c *_R X\ i\ \xi \le$ *cond-exp M* (*F i*) ($\lambda\xi.\ c *_R X\ j\ \xi$) ξ
using *cond-exp-scaleR-right*[*OF integrable*, *of i c j*] *supermartingale-property*[*OF asm*]

by (*auto intro*!: *scaleR-left-mono-neg*[*OF* - *assms*])
}
qed (*auto simp add*: *borel-measurable-integrable borel-measurable-scaleR integrable random-variable adapted borel-measurable-const-scaleR*)

lemma *uminus*[*intro*]:
shows *submartingale M F* $(- X)$
unfolding *fun-Compl-def* **using** *scaleR-nonpos*[*of* -1] **by** *simp*

lemma *min*:
assumes *supermartingale M F Y*
shows *supermartingale M F* $(\lambda i\ \xi.\ min\ (X\ i\ \xi)\ (Y\ i\ \xi))$
proof (*unfold-locales*)
interpret *Y*: *supermartingale M F Y* **by** (*rule assms*)
{
fix *i j* :: *'b* **assume** *asm*: $i \le j$
have *AE* ξ *in M*. $min\ (X\ i\ \xi)\ (Y\ i\ \xi) \ge min\ (cond\text{-}exp\ M\ (F\ i)\ (X\ j)\ \xi)\ (cond\text{-}exp\ M\ (F\ i)\ (Y\ j)\ \xi)$ **using** *supermartingale-property Y.supermartingale-property asm* **unfolding** *min-def* **by** *fastforce*
thus *AE* ξ *in M*. $min\ (X\ i\ \xi)\ (Y\ i\ \xi) \ge cond\text{-}exp\ M\ (F\ i)\ (\lambda\xi.\ min\ (X\ j\ \xi)\ (Y\ j\ \xi))\ \xi$ **using** *cond-exp-min*[*OF integrable Y.integrable*, *of i j j*] *order.trans* **by** *fast*
}
show $\bigwedge i.\ (\lambda\xi.\ min\ (X\ i\ \xi)\ (Y\ i\ \xi)) \in borel\text{-}measurable\ M\ \bigwedge i.\ (\lambda\xi.\ min\ (X\ i\ \xi)\ (Y\ i\ \xi)) \in borel\text{-}measurable\ (F\ i)\ \bigwedge i.\ integrable\ M\ (\lambda\xi.\ min\ (X\ i\ \xi)\ (Y\ i\ \xi))$ **by** (*force intro*: *Y.integrable integrable assms*)+
qed

lemma *min-0*:
shows *supermartingale M F* $(\lambda i\ \xi.\ min\ 0\ (X\ i\ \xi))$
proof −
interpret *zero*: *supermartingale M F* λ- -. *0* **by** (*intro martingale-order.is-supermartingale*, *unfold-locales*, *auto*)
show *?thesis* **by** (*intro zero.min supermartingale-axioms*)
qed

end

lemma (**in** *supermartingale-lattice*) *inf*:
assumes *supermartingale-lattice M F Y*
shows *supermartingale-lattice M F* $(\lambda i\ \xi.\ inf\ (X\ i\ \xi)\ (Y\ i\ \xi))$
using *supermartingale-lattice.intro supermartingale.min*[*OF supermartingale-axioms assms*[*THEN supermartingale-lattice.axioms*]] **unfolding** *inf-min*[*symmetric*] **.**

lemma (**in** *adapted-process-order*) *supermartingale-of-cond-exp-diff-nonneg*:
assumes *integrable*: $\bigwedge i.\ integrable\ M\ (X\ i)$
and *diff-nonneg*: $\bigwedge i\ j.\ i \le j \Longrightarrow AE\ x\ in\ M.\ 0 \le cond\text{-}exp\ M\ (F\ i)\ (\lambda\xi.\ X\ i\ \xi - X\ j\ \xi)\ x$
shows *supermartingale M F X*
proof

{
fix $i\ j :: 't$ **assume** *asm*: $i \le j$
show $AE\ \xi\ in\ M.\ X\ i\ \xi \ge cond\text{-}exp\ M\ (F\ i)\ (X\ j)\ \xi$
using *diff-nonneg*[*OF asm*] *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i i j*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] **by** *fastforce*
}
qed (*intro integrable*)

lemma (**in** *adapted-process-order*) *supermartingale-of-set-integral-ge*:
assumes *integrable*: $\bigwedge i.\ integrable\ M\ (X\ i)$
and $\bigwedge A\ i\ j.\ i \le j \Longrightarrow A \in F\ i \Longrightarrow set\text{-}lebesgue\text{-}integral\ M\ A\ (X\ j) \le set\text{-}lebesgue\text{-}integral\ M\ A\ (X\ i)$
shows *supermartingale M F X*
proof −
interpret *uminus-X*: *adapted-process-order M F* $-X$ **by** (*intro adapted-process-order.intro uminus*)
note $* =$ *set-integral-uminus*[*unfolded set-integrable-def*, *OF integrable-mult-indicator*[*OF - integrable*]]
have *supermartingale M F* $(-(-\ X))$ **using** *ord-eq-le-trans*[*OF* $*$ *ord-le-eq-trans*[*OF le-imp-neg-le*[*OF assms*(*2*)] $*$[*symmetric*]]] *subalg*
by (*intro submartingale.uminus uminus-X.submartingale-of-set-integral-le*) (*auto simp add*: *subalgebra-def integrable fun-Compl-def*, *blast*)
thus *?thesis* **unfolding** *fun-Compl-def* **by** *simp*
qed

2 Discrete Time Martingales

locale *discrete-time-martingale* = *martingale M F X* **for** *M F* **and** $X :: nat \Rightarrow - \Rightarrow -$
locale *discrete-time-submartingale* = *submartingale M F X* **for** *M F* **and** $X :: nat \Rightarrow - \Rightarrow -$
locale *discrete-time-supermartingale* = *supermartingale M F X* **for** *M F* **and** $X :: nat \Rightarrow - \Rightarrow -$

sublocale *discrete-time-martingale* $\subseteq$ *discrete-time-adapted-process* **by** (*unfold-locales*)
sublocale *discrete-time-submartingale* $\subseteq$ *discrete-time-adapted-process* **by** (*unfold-locales*)
sublocale *discrete-time-supermartingale* $\subseteq$ *discrete-time-adapted-process* **by** (*unfold-locales*)

3 Discrete Time Martingales

lemma (**in** *discrete-time-martingale*) *predictable-eq-bot*:
assumes *predictable X*
shows $AE\ \xi\ in\ M.\ X\ i\ \xi = X\ \bot\ \xi$
proof (*induction i*)
case *0*
then show *?case* **by** (*simp add*: *bot-nat-def*)
next
case (*Suc i*)

thus *?case* **using** *predictable-discrete-time-process-measurable*[*OF assms*, *of Suc i*]
martingale-property[*OF le-SucI*, *of i*]
cond-exp-F-meas[*OF integrable*, *of Suc i i*] *Suc* **by** *fastforce*
qed

lemma (**in** *discrete-time-adapted-process*) *martingale-of-set-integral-eq-Suc*:
assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
and $\bigwedge A\ i.\ A \in F\ i \Longrightarrow$ *set-lebesgue-integral M A* (*X i*) = *set-lebesgue-integral M A* (*X* (*Suc i*))
shows *discrete-time-martingale M F X*
proof (*intro discrete-time-martingale.intro martingale-of-set-integral-eq*)
fix *i j A* **assume** *asm*: $i \le j$ $A \in$ *sets* (*F i*)
show *set-lebesgue-integral M A* (*X i*) = *set-lebesgue-integral M A* (*X j*) **using** *asm*
proof (*induction* $j - i$ *arbitrary*: *i j*)
case *0*
then show *?case* **using** *asm* **by** *simp*
next
case (*Suc n*)
hence $*$: $n = j -$ *Suc i* **by** *linarith*
have *Suc i* $\le j$ **using** *Suc*(*2*,*3*) **by** *linarith*
thus *?case* **using** *sets-F-mono*[*OF le-SucI*] *Suc*(*4*) *Suc*(*1*)[*OF* $*$] **by** (*auto intro*: *assms*(*2*)[*THEN trans*])
qed
qed (*simp add*: *integrable*)

lemma (**in** *discrete-time-adapted-process*) *martingale-nat*:
assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
and $\bigwedge i$. *AE* ξ *in M*. *X i* ξ = *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ
shows *discrete-time-martingale M F X*
proof (*unfold-locales*)
fix *i j* :: *nat* **assume** *asm*: $i \le j$
show *AE* ξ *in M*. *X i* ξ = *cond-exp M* (*F i*) (*X j*) ξ **using** *asm*
proof (*induction* $j - i$ *arbitrary*: *i j*)
case *0*
hence $j = i$ **by** *simp*
thus *?case* **using** *cond-exp-F-meas*[*OF integrable adapted*, *THEN AE-symmetric*] **by** *presburger*
next
case (*Suc n*)
have *j*: $j =$ *Suc* $(n + i)$ **using** *Suc* **by** *linarith*
have *n*: $n = n + i - i$ **using** *Suc* **by** *linarith*
have $*$: *AE* ξ *in M*. *cond-exp M* (*F* $(n + i)$) (*X j*) ξ = *X* $(n + i)$ ξ **unfolding** *j* **using** *assms*(*2*)[*THEN AE-symmetric*] **by** *blast*
have *AE* ξ *in M*. *cond-exp M* (*F i*) (*X j*) ξ = *cond-exp M* (*F i*) (*cond-exp M* (*F* $(n + i)$) (*X j*)) ξ **by** (*intro cond-exp-nested-subalg integrable subalg*, *simp add*: *subalgebra-def space-F sets-F-mono*)
hence *AE* ξ *in M*. *cond-exp M* (*F i*) (*X j*) ξ = *cond-exp M* (*F i*) (*X* $(n + i)$)

ξ **using** *cond-exp-cong-AE*[*OF integrable-cond-exp integrable* $*$] **by** *force*
thus *?case* **using** *Suc*(*1*)[*OF n*] **by** *fastforce*
qed
qed (*simp add*: *integrable*)

lemma (**in** *discrete-time-adapted-process*) *martingale-of-cond-exp-diff-Suc-eq-0*:
assumes *integrable*: $\bigwedge i.$ *integrable M* (*X i*)
and $\bigwedge i.$ *AE* ξ *in M*. *0* = *cond-exp M* (*F i*) ($\lambda\xi$. *X* (*Suc i*) ξ − *X i* ξ) ξ
shows *discrete-time-martingale M F X*
proof (*intro martingale-nat integrable*)
fix *i*
show *AE* ξ *in M*. *X i* ξ = *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ **using** *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i Suc i i*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] *assms*(*2*)[*of i*] **by** *fastforce*
qed

4 Discrete Time Submartingales

lemma (**in** *discrete-time-submartingale*) *predictable-ge-bot*:
assumes *predictable X*
shows *AE* ξ *in M*. *X i* $\xi \geq$ *X* $\bot$ ξ
proof (*induction i*)
case *0*
then show *?case* **by** (*simp add*: *bot-nat-def*)
next
case (*Suc i*)
thus *?case* **using** *predictable-discrete-time-process-measurable*[*OF assms*, *of Suc i*]
submartingale-property[*OF le-SucI*, *of i*]
cond-exp-F-meas[*OF integrable*, *of Suc i i*] *Suc* **by** *fastforce*
qed

lemma (**in** *discrete-time-adapted-process-order*) *submartingale-of-set-integral-le-Suc*:
assumes *integrable*: $\bigwedge i.$ *integrable M* (*X i*)
and $\bigwedge A\ i.$ $A \in F\ i \Longrightarrow$ *set-lebesgue-integral M A* (*X i*) $\leq$ *set-lebesgue-integral M A* (*X* (*Suc i*))
shows *discrete-time-submartingale M F X*
proof (*intro discrete-time-submartingale.intro submartingale-of-set-integral-le*)
fix *i j A* **assume** *asm*: $i \leq j$ $A \in$ *sets* (*F i*)
show *set-lebesgue-integral M A* (*X i*) $\leq$ *set-lebesgue-integral M A* (*X j*) **using** *asm*
proof (*induction j* − *i arbitrary*: *i j*)
case *0*
then show *?case* **using** *asm* **by** *simp*
next
case (*Suc n*)
hence $*$: *n* = *j* − *Suc i* **by** *linarith*
have *Suc i* $\leq$ *j* **using** *Suc*(*2*,*3*) **by** *linarith*
thus *?case* **using** *sets-F-mono*[*OF le-SucI*] *Suc*(*4*) *Suc*(*1*)[*OF* $*$] **by** (*auto*

intro: *assms*(*2*)[*THEN order-trans*])
  qed
qed (*simp add*: *integrable*)

lemma (**in** *discrete-time-adapted-process-order*) *submartingale-nat*:
  assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
      and $\bigwedge i$. *AE* ξ *in M*. *X i* $\xi \leq$ *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ
    shows *discrete-time-submartingale M F X*
  using *subalg integrable assms*(*2*)
  by (*intro submartingale-of-set-integral-le-Suc ord-le-eq-trans*[*OF set-integral-mono-AE-banach cond-exp-set-integral*[*symmetric*]], *simp*)
        (*meson in-mono integrable-mult-indicator set-integrable-def subalgebra-def*,
      meson integrable-cond-exp in-mono integrable-mult-indicator set-integrable-def subalgebra-def,
         auto simp add: *subalgebra-def*, *metis* (*mono-tags*, *lifting*) *AE-I2 AE-mp*)

lemma (**in** *discrete-time-adapted-process-order*) *submartingale-of-cond-exp-diff-Suc-nonneg*:
  assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
      and $\bigwedge i$. *AE* ξ *in M*. *0* $\leq$ *cond-exp M* (*F i*) ($\lambda\xi$. *X* (*Suc i*) ξ − *X i* ξ) ξ
    shows *discrete-time-submartingale M F X*
proof (*intro submartingale-nat integrable*)
  fix *i*
  show *AE* ξ *in M*. *X i* $\xi \leq$ *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ **using** *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i Suc i i*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] *assms*(*2*)[*of i*] **by** *fastforce*
qed

5 Discrete Time Supermartingales

lemma (**in** *discrete-time-supermartingale*) *predictable-le-bot*:
  assumes *predictable X*
  shows *AE* ξ *in M*. *X i* $\xi \leq X \perp \xi$
proof (*induction i*)
  case *0*
  then show *?case* **by** (*simp add*: *bot-nat-def*)
next
  case (*Suc i*)
  thus *?case* **using** *predictable-discrete-time-process-measurable*[*OF assms*, *of Suc i*]
          supermartingale-property[*OF le-SucI*, *of i*]
          cond-exp-F-meas[*OF integrable*, *of Suc i i*] *Suc* **by** *fastforce*
qed

lemma (**in** *discrete-time-adapted-process-order*) *supermartingale-of-set-integral-ge-Suc*:
  assumes *integrable*: $\bigwedge i$. *integrable M* (*X i*)
    and $\bigwedge A\ i$. $A \in F\ i \Longrightarrow$ *set-lebesgue-integral M A* (*X* (*Suc i*)) $\leq$ *set-lebesgue-integral M A* (*X i*)
    shows *discrete-time-supermartingale M F X*
proof −

interpret *uminus-X*: *discrete-time-adapted-process-order M F −X* **by** (*intro discrete-time-adapted-process-order.intro adapted-process-order.intro uminus*)
note ∗ = *set-integral-uminus*[*unfolded set-integrable-def*, *OF integrable-mult-indicator*[*OF - integrable*]]
have *discrete-time-supermartingale M F* (−(− *X*)) **using** *ord-eq-le-trans*[*OF* ∗ *ord-le-eq-trans*[*OF le-imp-neg-le*[*OF assms*(*2*)] ∗[*symmetric*]]] *subalg*
by (*intro discrete-time-supermartingale.intro submartingale.uminus discrete-time-submartingale.axioms uminus-X.submartingale-of-set-integral-le-Suc*) (*auto simp add*: *subalgebra-def integrable fun-Compl-def*, *blast*)
thus *?thesis* **unfolding** *fun-Compl-def* **by** *simp*
qed

lemma (**in** *discrete-time-adapted-process-order*) *supermartingale-nat*:
assumes *integrable*: $\bigwedge i.$ *integrable M* (*X i*)
and $\bigwedge i.$ *AE* ξ *in M*. *X i* $\xi \geq$ *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ
shows *discrete-time-supermartingale M F X*
proof −
interpret *uminus-X*: *discrete-time-adapted-process-order M F −X* **by** (*intro discrete-time-adapted-process-order.intro adapted-process-order.intro uminus*)
have *AE* ξ *in M*. − *X i* $\xi \leq$ *cond-exp M* (*F i*) ($\lambda x.$ − *X* (*Suc i*) *x*) ξ **for** *i* **using** *assms*(*2*) *cond-exp-uminus*[*OF integrable*, *of i Suc i*] **by** *force*
hence *discrete-time-supermartingale M F* (−(− *X*)) **by** (*intro discrete-time-supermartingale.intro submartingale.uminus discrete-time-submartingale.axioms uminus-X.submartingale-nat*) (*simp only*: *fun-Compl-def*, *intro integrable-minus integrable*, *auto simp add*: *fun-Compl-def*)
thus *?thesis* **unfolding** *fun-Compl-def* **by** *simp*
qed

lemma (**in** *discrete-time-adapted-process-order*) *supermartingale-of-cond-exp-diff-Suc-nonneg*:
assumes *integrable*: $\bigwedge i.$ *integrable M* (*X i*)
and $\bigwedge i.$ *AE* ξ *in M*. *0* $\leq$ *cond-exp M* (*F i*) ($\lambda\xi.$ *X i* ξ − *X* (*Suc i*) ξ) ξ
shows *discrete-time-supermartingale M F X*
proof (*intro supermartingale-nat integrable*)
fix *i*
show *AE* ξ *in M*. *X i* $\xi \geq$ *cond-exp M* (*F i*) (*X* (*Suc i*)) ξ **using** *cond-exp-diff*[*OF integrable*(*1*,*1*), *of i i Suc i*] *cond-exp-F-meas*[*OF integrable adapted*, *of i*] *assms*(*2*)[*of i*] **by** *fastforce*
qed

end

References

[1] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a Cryptographic Perspective.* Springer US, Boston, MA, 2002. OCLC: 852791069.