The Hurwitz and Riemann ζ functions

Manuel Eberl

December 14, 2021

Abstract

This entry builds upon the results about formal and analytic Dirichlet series to define the Hurwitz ζ function $\zeta(a, s)$ and, based on that, the Riemann ζ function $\zeta(s)$. This is done by first defining them for $\mathfrak{R}(z) > 1$ and then successively extending the domain to the left using the Euler–MacLaurin formula.

Apart from the most basic facts such as analyticity, the following results are provided:

- the Stieltjes constants and the Laurent expansion of $\zeta(s)$ at $s = 1$
- the non-vanishing of $\zeta(s)$ for $\mathfrak{R}(s) \geq 1$
- the relationship between $\zeta(a, s)$ and Γ
- the special values at negative integers and positive even integers
- Hurwitz's formula and the reflection formula for $\zeta(s)$
- the Hadjicostas–Chapman formula [3, 4]

The entry also contains Euler's analytic proof of the infinitude of primes, based on the fact that $\zeta(s)$ has a pole at $s = 1$.

Contents

1 Various preliminary material

theory *Zeta-Library*
imports
HOL−Complex-Analysis.Complex-Analysis
HOL−Real-Asymp.Real-Asymp
Dirichlet-Series.Dirichlet-Series-Analysis
begin

1.1 Facts about limits

lemma *at-within-altdef*:
$at\ x\ within\ A = (INF\ S{\in}\{S.\ open\ S \wedge x \in S\}.\ principal\ (S \cap (A - \{x\})))$
unfolding *at-within-def nhds-def inf-principal* [*symmetric*]
by (*subst INF-inf-distrib* [*symmetric*]) (*auto simp*: *INF-constant*)

lemma *tendsto-at-left-realI-sequentially*:
fixes $f :: real \Rightarrow 'b{::}first\text{-}countable\text{-}topology$
assumes $*$: $\bigwedge X.\ filterlim\ X\ (at\text{-}left\ c)\ sequentially \Longrightarrow (\lambda n.\ f\ (X\ n)) \longrightarrow y$
shows $(f \longrightarrow y)\ (at\text{-}left\ c)$
proof −
obtain A **where** A: $decseq\ A\ open\ (A\ n)\ y \in A\ n\ nhds\ y = (INF\ n.\ principal\ (A\ n))$ **for** n
by (*rule nhds-countable*[*of y*]) (*rule that*)

have $\forall m.\ \exists d<c.\ \forall x{\in}\{d<..<c\}.\ f\ x \in A\ m$
proof (*rule ccontr*)
assume $\neg\ (\forall m.\ \exists d<c.\ \forall x{\in}\{d<..<c\}.\ f\ x \in A\ m)$
then obtain m **where** $**$: $\bigwedge d.\ d < c \Longrightarrow \exists x{\in}\{d<..<c\}.\ f\ x \notin A\ m$
by *auto*
have $\exists X.\ \forall n.\ (f\ (X\ n) \notin A\ m \wedge X\ n < c) \wedge X\ (Suc\ n) > c - max\ 0\ ((c - X\ n)\ /\ 2)$
proof (*intro dependent-nat-choice*, *goal-cases*)
case *1*
from $**$[*of* $c - 1$] **show** *?case* **by** *auto*
next
case (*2 x n*)
with $**$[*of* $c - max\ 0\ (c - x)\ /\ 2$] **show** *?case* **by** *force*
qed
then obtain X **where** X: $\bigwedge n.\ f\ (X\ n) \notin A\ m\ \bigwedge n.\ X\ n < c\ \bigwedge n.\ X\ (Suc\ n) > c - max\ 0\ ((c - X\ n)\ /\ 2)$
by *auto* (*metis diff-gt-0-iff-gt half-gt-zero-iff max.absorb3 max.commute*)
have *X-ge*: $X\ n \geq c - (c - X\ 0)\ /\ 2\ \hat{}\ n$ **for** n
proof (*induction n*)
case (*Suc n*)
have $c - (c - X\ 0)\ /\ 2\ \hat{}\ Suc\ n = c - (c - (c - (c - X\ 0)\ /\ 2\ \hat{}\ n))\ /\ 2$
by *simp*
also have $c - (c - (c - (c - X\ 0)\ /\ 2\ \hat{}\ n))\ /\ 2 \leq c - (c - X\ n)\ /\ 2$
by (*intro diff-left-mono divide-right-mono Suc diff-right-mono*) *auto*
also have $\ldots = c - max\ 0\ ((c - X\ n)\ /\ 2)$

using $X[of\ n]$ **by** (*simp add*: *max-def*)
also have ... $< X$ (*Suc n*)
using $X[of\ n]$ **by** *simp*
finally show *?case* **by** *linarith*
qed *auto*

have $X \longrightarrow c$
proof (*rule tendsto-sandwich*)
show *eventually* ($\lambda n.\ X\ n \leq c$) *sequentially*
using X **by** (*intro always-eventually*) (*auto intro*!: *less-imp-le*)
show *eventually* ($\lambda n.\ X\ n \geq c - (c - X\ 0)\ /\ 2\ \widehat{}\ n$) *sequentially*
using *X-ge* **by** (*intro always-eventually*) *auto*
qed *real-asymp+*
hence *filterlim X* (*at-left c*) *sequentially*
by (*rule tendsto-imp-filterlim-at-left*)
(*use X* **in** ‹*auto intro*!: *always-eventually less-imp-le*›)
from *topological-tendstoD*[*OF* ∗[*OF this*] *A*(*2*, *3*), *of m*] *X*(*1*) **show** *False*
by *auto*
qed

then obtain *d* **where** *d*: $d\ m < c\ x \in \{d\ m{<}..{<}c\} \Longrightarrow f\ x \in A\ m$ **for** *m x*
by *metis*
have ∗∗∗: *at-left c* = (*INF* $S \in \{S.\ open\ S \wedge c \in S\}$. *principal* ($S \cap \{..{<}c\}$))
by (*simp add*: *at-within-altdef*)
from *d* **show** *?thesis*
unfolding ∗∗∗ *A* **using** *A*(*1*,*2*) **by** (*intro filterlim-base*[*of* - $\lambda m.\ \{d\ m{<}..\}$])
auto
qed

lemma
shows *at-right-PInf* [*simp*]: *at-right* (∞ :: *ereal*) = *bot*
and *at-left-MInf* [*simp*]: *at-left* ($-\infty$ :: *ereal*) = *bot*
proof −
have $\{(\infty{::}ereal){<}..\} = \{\}$ $\{..{<}-(\infty{::}ereal)\} = \{\}$
by *auto*
thus *at-right* (∞ :: *ereal*) = *bot at-left* ($-\infty$ :: *ereal*) = *bot*
by (*simp-all add*: *at-within-def*)
qed

lemma *tendsto-at-left-erealI-sequentially*:
fixes *f* :: *ereal* ⇒ ′*b*::*first-countable-topology*
assumes ∗: $\bigwedge X.$ *filterlim X* (*at-left c*) *sequentially* $\Longrightarrow$ ($\lambda n.\ f\ (X\ n)$) $\longrightarrow y$
shows ($f \longrightarrow y$) (*at-left c*)
proof (*cases c*)
case [*simp*]: *PInf*
have (($\lambda x.\ f$ (*ereal x*)) $\longrightarrow y$) *at-top* **using** *assms*
by (*intro tendsto-at-topI-sequentially assms*)
(*simp-all flip*: *ereal-tendsto-simps add*: *o-def filterlim-at*)
thus *?thesis*

by (*simp add*: *at-left-PInf filterlim-filtermap*)
next
case [*simp*]: *MInf*
thus *?thesis* **by** *auto*
next
case [*simp*]: (*real c'*)
have ((λx. *f* (*ereal x*)) $\longrightarrow$ *y*) (*at-left c'*)
proof (*intro tendsto-at-left-realI-sequentially assms*)
fix *X* **assume** $*$: *filterlim X* (*at-left c'*) *sequentially*
show *filterlim* (λn. *ereal* (*X n*)) (*at-left c*) *sequentially*
by (*rule filterlim-compose*[*OF* - $*$])
(*simp add*: *sequentially-imp-eventually-within tendsto-imp-filterlim-at-left*)
qed
thus *?thesis*
by (*simp add*: *at-left-ereal filterlim-filtermap*)
qed

lemma *tendsto-at-right-realI-sequentially*:
fixes *f* :: *real* $\Rightarrow$ *'b::first-countable-topology*
assumes $*$: $\bigwedge X$. *filterlim X* (*at-right c*) *sequentially* $\Longrightarrow$ (λn. *f* (*X n*)) $\longrightarrow$ *y*
shows (*f* $\longrightarrow$ *y*) (*at-right c*)
proof $-$
obtain *A* **where** *A*: *decseq A open* (*A n*) *y* $\in$ *A n nhds y* = (*INF n. principal* (*A n*)) **for** *n*
by (*rule nhds-countable*[*of y*]) (*rule that*)

have $\forall m. \exists d>c. \forall x \in \{c<..<d\}$. *f x* $\in$ *A m*
proof (*rule ccontr*)
assume $\neg$ ($\forall m. \exists d>c. \forall x \in \{c<..<d\}$. *f x* $\in$ *A m*)
then obtain *m* **where** $**$: $\bigwedge d$. $d > c \Longrightarrow \exists x \in \{c<..<d\}$. *f x* $\notin$ *A m*
by *auto*
have $\exists X. \forall n$. (*f* (*X n*) $\notin$ *A m* $\wedge$ *X n* $>$ *c*) $\wedge$ *X* (*Suc n*) $<$ *c* + *max 0* ((*X n* $-$ *c*) / *2*)
proof (*intro dependent-nat-choice*, *goal-cases*)
case *1*
from $**$[*of c* + *1*] **show** *?case* **by** *auto*
next
case (*2 x n*)
with $**$[*of c* + *max 0* (*x* $-$ *c*) / *2*] **show** *?case* **by** *force*
qed
then obtain *X* **where** *X*: $\bigwedge n$. *f* (*X n*) $\notin$ *A m* $\bigwedge n$. *X n* $>$ *c* $\bigwedge n$. *X* (*Suc n*) $<$ *c* + *max 0* ((*X n* $-$ *c*) / *2*)
by *auto* (*metis add.left-neutral half-gt-zero-iff less-diff-eq max.absorb4*)
have *X-le*: *X n* $\leq$ *c* + (*X 0* $-$ *c*) / *2* ^ *n* **for** *n*
proof (*induction n*)
case (*Suc n*)
have *X* (*Suc n*) $<$ *c* + *max 0* ((*X n* $-$ *c*) / *2*)
by (*intro X*)
also have ... = *c* + (*X n* $-$ *c*) / *2*

```
      using X[of n] by (simp add: field-simps max-def)
    also have ... ≤ c + (c + (X 0 − c) / 2 ^ n − c) / 2
      by (intro add-left-mono divide-right-mono Suc diff-right-mono) auto
    also have ... = c + (X 0 − c) / 2 ^ Suc n
      by simp
    finally show ?case by linarith
  qed auto

  have X ⟶ c
  proof (rule tendsto-sandwich)
    show eventually (λn. X n ≥ c) sequentially
      using X by (intro always-eventually) (auto intro!: less-imp-le)
    show eventually (λn. X n ≤ c + (X 0 − c) / 2 ^ n) sequentially
      using X-le by (intro always-eventually) auto
  qed real-asymp+
  hence filterlim X (at-right c) sequentially
    by (rule tendsto-imp-filterlim-at-right)
       (use X in ‹auto intro!: always-eventually less-imp-le›)
  from topological-tendstoD[OF ∗[OF this] A(2, 3), of m] X(1) show False
    by auto
qed

then obtain d where d: d m > c x ∈ {c<..<d m} ⟹ f x ∈ A m for m x
  by metis
have ∗∗∗: at-right c = (INF S∈{S. open S ∧ c ∈ S}. principal (S ∩ {c<..}))
  by (simp add: at-within-altdef)
from d show ?thesis
   unfolding ∗∗∗ A using A(1,2) by (intro filterlim-base[of - λm. {..<d m}])
auto
qed

lemma tendsto-at-right-erealI-sequentially:
  fixes f :: ereal ⇒ 'b::first-countable-topology
  assumes ∗: ⋀X. filterlim X (at-right c) sequentially ⟹ (λn. f (X n)) ⟶ y
  shows (f ⟶ y) (at-right c)
proof (cases c)
  case [simp]: MInf
  have ((λx. f (−ereal x)) ⟶ y) at-top using assms
   by (intro tendsto-at-topI-sequentially assms)
      (simp-all flip: uminus-ereal.simps ereal-tendsto-simps add: o-def filterlim-at)
  thus ?thesis
    by (simp add: at-right-MInf filterlim-filtermap at-top-mirror)
next
  case [simp]: PInf
  thus ?thesis by auto
next
  case [simp]: (real c')
  have ((λx. f (ereal x)) ⟶ y) (at-right c')
  proof (intro tendsto-at-right-realI-sequentially assms)
```

fix *X* **assume** ∗: *filterlim X* (*at-right c′*) *sequentially*
show *filterlim* (*λn. ereal* (*X n*)) (*at-right c*) *sequentially*
by (*rule filterlim-compose*[*OF* - ∗])
(*simp add*: *sequentially-imp-eventually-within tendsto-imp-filterlim-at-right*)
qed
thus *?thesis*
by (*simp add*: *at-right-ereal filterlim-filtermap*)
qed

proposition *analytic-continuation′*:
assumes *hol*: *f holomorphic-on S g holomorphic-on S*
and *open S* **and** *connected S*
and *U* ⊆ *S* **and** *ξ* ∈ *S*
and *ξ islimpt U*
and *fU0* [*simp*]: $\bigwedge z.\ z \in U \Longrightarrow f\ z = g\ z$
and *w* ∈ *S*
shows *f w* = *g w*
using *analytic-continuation*[*OF holomorphic-on-diff*[*OF hol*] *assms*(*3*−*7*) - *assms*(*9*)] *assms*(*8*)
by *simp*

1.2 Various facts about integrals

lemma *continuous-on-imp-set-integrable-cbox*:
fixes *h* :: *′a* :: *euclidean-space* ⇒ *′b* :: *euclidean-space*
assumes *continuous-on* (*cbox a b*) *h*
shows *set-integrable lborel* (*cbox a b*) *h*
proof −
from *assms* **have** *h absolutely-integrable-on cbox a b*
by (*rule absolutely-integrable-continuous*)
moreover have (*λx. indicat-real* (*cbox a b*) *x* $*_R$ *h x*) ∈ *borel-measurable borel*
by (*rule borel-measurable-continuous-on-indicator*) (*use assms* **in** *auto*)
ultimately show *?thesis*
unfolding *set-integrable-def* **using** *assms* **by** (*subst* (*asm*) *integrable-completion*) *auto*
qed

lemma *set-nn-integral-cong*:
assumes *M* = *M′ A* = *B* $\bigwedge x.\ x \in$ *space M* ∩ *A* ⟹ *f x* = *g x*
shows *set-nn-integral M A f* = *set-nn-integral M′ B g*
using *assms* **unfolding** *assms*(*1*,*2*) **by** (*intro nn-integral-cong*) (*auto simp*: *indicator-def*)

lemma *set-integrable-bound*:
fixes *f* :: *′a* ⇒ *′b*::{*banach*, *second-countable-topology*}
and *g* :: *′a* ⇒ *′c*::{*banach*, *second-countable-topology*}
assumes *set-integrable M A f set-borel-measurable M A g*
assumes *AE x in M*. *x* ∈ *A* ⟶ *norm* (*g x*) ≤ *norm* (*f x*)
shows *set-integrable M A g*

unfolding *set-integrable-def*
proof (*rule Bochner-Integration.integrable-bound*)
from *assms*(*1*) **show** *integrable M* (λx. *indicator A x* $*_R$ *f x*)
by (*simp add*: *set-integrable-def*)
from *assms*(*2*) **show** (λx. *indicat-real A x* $*_R$ *g x*) $\in$ *borel-measurable M*
by (*simp add*: *set-borel-measurable-def*)
from *assms*(*3*) **show** *AE x in M*. *norm* (*indicat-real A x* $*_R$ *g x*) $\leq$ *norm* (*indicat-real A x* $*_R$ *f x*)
by *eventually-elim* (*simp add*: *indicator-def*)
qed

lemma *nn-integral-has-integral-lebesgue*:
fixes *f* :: *'a::euclidean-space* $\Rightarrow$ *real*
assumes *nonneg*: $\bigwedge x$. $x \in \Omega \Longrightarrow 0 \leq f\ x$ **and** *I*: (*f has-integral I*) Ω
shows *integral*^{N} *lborel* (λx. *indicator* Ω *x* $*$ *f x*) = *I*
proof –
from *I* **have** (λx. *indicator* Ω *x* $*_R$ *f x*) $\in$ *lebesgue* $\rightarrow_M$ *borel*
by (*rule has-integral-implies-lebesgue-measurable*)
then obtain *f'* :: *'a* $\Rightarrow$ *real*
where [*measurable*]: *f'* $\in$ *borel* $\rightarrow_M$ *borel* **and** *eq*: *AE x in lborel*. *indicator* Ω *x* $*$ *f x* = *f' x*
by (*auto dest*: *completion-ex-borel-measurable-real*)

from *I* **have** ((λx. *abs* (*indicator* Ω *x* $*$ *f x*)) *has-integral I*) *UNIV*
using *nonneg* **by** (*simp add*: *indicator-def of-bool-def if-distrib*[*of* λx. *x* $*$ *f y* **for** *y*] *cong*: *if-cong*)
also have ((λx. *abs* (*indicator* Ω *x* $*$ *f x*)) *has-integral I*) *UNIV* $\longleftrightarrow$ ((λx. *abs* (*f' x*)) *has-integral I*) *UNIV*
using *eq* **by** (*intro has-integral-AE*) *auto*
finally have *integral*^{N} *lborel* (λx. *abs* (*f' x*)) = *I*
by (*rule nn-integral-has-integral-lborel*[*rotated 2*]) *auto*
also have *integral*^{N} *lborel* (λx. *abs* (*f' x*)) = *integral*^{N} *lborel* (λx. *abs* (*indicator* Ω *x* $*$ *f x*))
using *eq* **by** (*intro nn-integral-cong-AE*) *auto*
also have (λx. *abs* (*indicator* Ω *x* $*$ *f x*)) = (λx. *indicator* Ω *x* $*$ *f x*)
using *nonneg* **by** (*auto simp*: *indicator-def fun-eq-iff*)
finally show *?thesis* **.**
qed

1.3 Uniform convergence of integrals

lemma *has-absolute-integral-change-of-variables-1'*:
fixes *f* :: *real* $\Rightarrow$ *real* **and** *g* :: *real* $\Rightarrow$ *real*
assumes *S*: *S* $\in$ *sets lebesgue*
and *der-g*: $\bigwedge x$. $x \in S \Longrightarrow$ (*g has-field-derivative g' x*) (*at x within S*)
and *inj*: *inj-on g S*
shows (λx. $|g'\ x|$ $*_R$ *f*(*g x*)) *absolutely-integrable-on S* $\wedge$
integral S (λx. $|g'\ x|$ $*_R$ *f*(*g x*)) = *b*

$\longleftrightarrow$ *f absolutely-integrable-on* (*g* ‘ *S*) $\wedge$ *integral* (*g* ‘ *S*) *f* = *b*
proof −
have ($\lambda x.$ $|g'\ x|$ $*_R$ *vec* (*f*(*g x*)) :: *real* ^ *1*) *absolutely-integrable-on S* $\wedge$
integral S ($\lambda x.$ $|g'\ x|$ $*_R$ *vec* (*f*(*g x*))) = (*vec b* :: *real* ^ *1*)
$\longleftrightarrow$ ($\lambda x.$ *vec* (*f x*) :: *real* ^ *1*) *absolutely-integrable-on* (*g* ‘ *S*) $\wedge$
integral (*g* ‘ *S*) ($\lambda x.$ *vec* (*f x*)) = (*vec b* :: *real* ^ *1*)
using *assms* **unfolding** *has-field-derivative-iff-has-vector-derivative*
by (*intro has-absolute-integral-change-of-variables-1 assms*) *auto*
thus *?thesis*
by (*simp add*: *absolutely-integrable-on-1-iff integral-on-1-eq*)
qed

lemma *set-nn-integral-lborel-eq-integral*:
fixes *f*::′*a*::*euclidean-space* $\Rightarrow$ *real*
assumes *set-borel-measurable borel A f*
assumes $\bigwedge x.$ $x \in A \Longrightarrow 0 \le f\ x$ ($\int^+ x{\in}A.\ f\ x\ \partial lborel$) $< \infty$
shows ($\int^+ x{\in}A.\ f\ x\ \partial lborel$) = *integral A f*
proof −
have *eq*: ($\int^+ x{\in}A.\ f\ x\ \partial lborel$) = ($\int^+ x.$ *ennreal* (*indicator A x* $*$ *f x*) $\partial lborel$)
by (*intro nn-integral-cong*) (*auto simp*: *indicator-def*)
also have ... = *integral UNIV* ($\lambda x.$ *indicator A x* $*$ *f x*)
using *assms eq* **by** (*intro nn-integral-lborel-eq-integral*)
(*auto simp*: *indicator-def set-borel-measurable-def*)
also have *integral UNIV* ($\lambda x.$ *indicator A x* $*$ *f x*) = *integral A* ($\lambda x.$ *indicator A x* $*$ *f x*)
by (*rule integral-spike-set*) (*auto intro*: *empty-imp-negligible*)

also have ... = *integral A f*
by (*rule integral-cong*) (*auto simp*: *indicator-def*)
finally show *?thesis* **.**
qed

lemma *nn-integral-has-integral-lebesgue′*:
fixes *f* :: ′*a*::*euclidean-space* $\Rightarrow$ *real*
assumes *nonneg*: $\bigwedge x.$ $x \in \Omega \Longrightarrow 0 \le f\ x$ **and** *I*: (*f has-integral I*) Ω
shows *integral*^{N} *lborel* ($\lambda x.$ *ennreal* (*f x*) $*$ *indicator* Ω *x*) = *ennreal I*
proof −
have *integral*^{N} *lborel* ($\lambda x.$ *ennreal* (*f x*) $*$ *indicator* Ω *x*) =
integral^N *lborel* ($\lambda x.$ *ennreal* (*indicator* Ω *x* $*$ *f x*))
by (*intro nn-integral-cong*) (*auto simp*: *indicator-def*)
also have ... = *ennreal I*
using *assms* **by** (*intro nn-integral-has-integral-lebesgue*)
finally show *?thesis* **.**
qed

lemma *uniform-limit-set-lebesgue-integral*:
fixes *f* :: ′*a* $\Rightarrow$ ′*b* :: *euclidean-space* $\Rightarrow$ ′*c* :: {*banach*, *second-countable-topology*}
assumes *set-integrable lborel X′ g*
assumes [*measurable*]: *X′* $\in$ *sets borel*

```
  assumes [measurable]: ⋀y. y ∈ Y ⟹ set-borel-measurable borel X′ (f y)
  assumes ⋀y. y ∈ Y ⟹ (AE t∈X′ in lborel. norm (f y t) ≤ g t)
  assumes eventually (λx. X x ∈ sets borel ∧ X x ⊆ X′) F
  assumes filterlim (λx. set-lebesgue-integral lborel (X x) g)
           (nhds (set-lebesgue-integral lborel X′ g)) F
  shows uniform-limit Y
          (λx y. set-lebesgue-integral lborel (X x) (f y))
          (λy. set-lebesgue-integral lborel X′ (f y)) F
proof (rule uniform-limitI, goal-cases)
  case (1 ε)
  have integrable-g: set-integrable lborel U g
    if U ∈ sets borel U ⊆ X′ for U
    by (rule set-integrable-subset[OF assms(1)]) (use that in auto)
  have eventually (λx. dist (set-lebesgue-integral lborel (X x) g)
                        (set-lebesgue-integral lborel X′ g) < ε) F
    using ‹ε > 0› assms by (auto simp: tendsto-iff)
  from this show ?case using ‹eventually (λ-. - ∧ -) F›
  proof eventually-elim
    case (elim x)
    hence [measurable]:X x ∈ sets borel and X x ⊆ X′ by auto
    have integrable: set-integrable lborel U (f y)
      if y ∈ Y U ∈ sets borel U ⊆ X′ for y U
      apply (rule set-integrable-subset)
        apply (rule set-integrable-bound[OF assms(1)])
         apply (use assms(3) that in ‹simp add: set-borel-measurable-def›)
      using assms(4)[OF ‹y ∈ Y›] apply eventually-elim apply force
      using that apply simp-all
      done
    show ?case
    proof
      fix y assume y ∈ Y
      have dist (set-lebesgue-integral lborel (X x) (f y))
              (set-lebesgue-integral lborel X′ (f y)) =
           norm (set-lebesgue-integral lborel X′ (f y) −
                set-lebesgue-integral lborel (X x) (f y))
        by (simp add: dist-norm norm-minus-commute)
      also have set-lebesgue-integral lborel X′ (f y) −
                  set-lebesgue-integral lborel (X x) (f y) =
                set-lebesgue-integral lborel (X′ − X x) (f y)
        unfolding set-lebesgue-integral-def
        apply (subst Bochner-Integration.integral-diff [symmetric])
        unfolding set-integrable-def [symmetric]
          apply (rule integrable; (fact | simp))
         apply (rule integrable; fact)
        apply (intro Bochner-Integration.integral-cong)
         apply (use ‹X x ⊆ X′› in ‹auto simp: indicator-def›)
        done
      also have norm ... ≤ (∫ t∈X′−X x. norm (f y t) ∂lborel)
        by (intro set-integral-norm-bound integrable) (fact | simp)+
```

also have $AE\ t{\in}X' - X\ x\ in\ lborel.\ norm\ (f\ y\ t) \leq g\ t$
using $assms(4)[OF\ \langle y \in Y\rangle]$ **by** *eventually-elim auto*
with $\langle y \in Y\rangle$ **have** $(\int t{\in}X'{-}X\ x.\ norm\ (f\ y\ t)\ \partial lborel) \leq (\int t{\in}X'{-}X\ x.\ g\ t\ \partial lborel)$
by (*intro set-integral-mono-AE set-integrable-norm integrable integrable-g*) *auto*
also have $\ldots = (\int t{\in}X'.\ g\ t\ \partial lborel) - (\int t{\in}X\ x.\ g\ t\ \partial lborel)$
unfolding *set-lebesgue-integral-def*
apply (*subst Bochner-Integration.integral-diff* [*symmetric*])
unfolding *set-integrable-def* [*symmetric*]
apply (*rule integrable-g*; (*fact* | *simp*))
apply (*rule integrable-g*; *fact*)
apply (*intro Bochner-Integration.integral-cong*)
apply (*use* $\langle X\ x \subseteq X'\rangle$ **in** ‹*auto simp*: *indicator-def*›)
done
also have $\ldots \leq dist\ (\int t{\in}X\ x.\ g\ t\ \partial lborel)\ (\int t{\in}X'.\ g\ t\ \partial lborel)$
by (*simp add*: *dist-norm*)
also have $\ldots < \varepsilon$ **by** *fact*
finally show *dist* (*set-lebesgue-integral lborel* $(X\ x)$ $(f\ y)$)
(*set-lebesgue-integral lborel* X' $(f\ y)) < \varepsilon$ **.**
qed
qed
qed

lemma *integral-dominated-convergence-at-right*:
fixes $s :: real \Rightarrow 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$ **and** $w :: 'a \Rightarrow real$
and $f :: 'a \Rightarrow 'b$ **and** M **and** $c :: real$
assumes $f \in borel\text{-}measurable\ M\ \bigwedge t.\ s\ t \in borel\text{-}measurable\ M\ integrable\ M\ w$
assumes *lim*: $AE\ x\ in\ M.\ ((\lambda i.\ s\ i\ x) \longrightarrow f\ x)\ (at\text{-}right\ c)$
assumes *bound*: $\forall_F\ i\ in\ at\text{-}right\ c.\ AE\ x\ in\ M.\ norm\ (s\ i\ x) \leq w\ x$
shows $((\lambda t.\ integral^L\ M\ (s\ t)) \longrightarrow integral^L\ M\ f)\ (at\text{-}right\ c)$
proof (*rule tendsto-at-right-realI-sequentially*)
fix $X :: nat \Rightarrow real$ **assume** X: *filterlim* X (*at-right* c) *sequentially*
from *filterlim-iff*[*THEN iffD1*, *OF this*, *rule-format*, *OF bound*]
obtain N **where** w: $\bigwedge n.\ N \leq n \Longrightarrow AE\ x\ in\ M.\ norm\ (s\ (X\ n)\ x) \leq w\ x$
by (*auto simp*: *eventually-sequentially*)

show $(\lambda n.\ integral^L\ M\ (s\ (X\ n))) \longrightarrow integral^L\ M\ f$
proof (*rule LIMSEQ-offset*, *rule integral-dominated-convergence*)
show $AE\ x\ in\ M.\ norm\ (s\ (X\ (n + N))\ x) \leq w\ x$ **for** n
by (*rule w*) *auto*
show $AE\ x\ in\ M.\ (\lambda n.\ s\ (X\ (n + N))\ x) \longrightarrow f\ x$
using *lim*
proof *eventually-elim*
fix x **assume** $((\lambda i.\ s\ i\ x) \longrightarrow f\ x)\ (at\text{-}right\ c)$
then show $(\lambda n.\ s\ (X\ (n + N))\ x) \longrightarrow f\ x$
by (*intro LIMSEQ-ignore-initial-segment filterlim-compose*[*OF* - X])
qed
qed *fact+*

qed

lemma *integral-dominated-convergence-at-left*:
  fixes s :: *real* $\Rightarrow$ $'a$ $\Rightarrow$ $'b$::{*banach*, *second-countable-topology*} **and** w :: $'a$ $\Rightarrow$ *real*
    and f :: $'a$ $\Rightarrow$ $'b$ **and** M **and** c :: *real*
  assumes $f \in$ *borel-measurable* M $\bigwedge t.\ s\ t \in$ *borel-measurable* M *integrable* M w
  assumes *lim*: *AE* x *in* M. $((\lambda i.\ s\ i\ x) \longrightarrow f\ x)$ (*at-left* c)
  assumes *bound*: $\forall_F$ i *in at-left* c. *AE* x *in* M. *norm* $(s\ i\ x) \le w\ x$
  shows $((\lambda t.$ *integral*^{L} M $(s\ t)) \longrightarrow$ *integral*^{L} M $f)$ (*at-left* c)
proof (*rule tendsto-at-left-realI-sequentially*)
  fix X :: *nat* $\Rightarrow$ *real* **assume** X: *filterlim* X (*at-left* c) *sequentially*
  from *filterlim-iff*[*THEN iffD1*, *OF this*, *rule-format*, *OF bound*]
  obtain N **where** w: $\bigwedge n.\ N \le n \Longrightarrow$ *AE* x *in* M. *norm* $(s\ (X\ n)\ x) \le w\ x$
    by (*auto simp*: *eventually-sequentially*)

  show $(\lambda n.$ *integral*^{L} M $(s\ (X\ n))) \longrightarrow$ *integral*^{L} M f
  proof (*rule LIMSEQ-offset*, *rule integral-dominated-convergence*)
    show *AE* x *in* M. *norm* $(s\ (X\ (n + N))\ x) \le w\ x$ **for** n
      by (*rule* w) *auto*
    show *AE* x *in* M. $(\lambda n.\ s\ (X\ (n + N))\ x) \longrightarrow f\ x$
      using *lim*
    proof *eventually-elim*
      fix x **assume** $((\lambda i.\ s\ i\ x) \longrightarrow f\ x)$ (*at-left* c)
      then show $(\lambda n.\ s\ (X\ (n + N))\ x) \longrightarrow f\ x$
        by (*intro LIMSEQ-ignore-initial-segment filterlim-compose*[*OF - X*])
    qed
  qed *fact+*
qed

lemma *uniform-limit-interval-integral-right*:
  fixes f :: $'a$ $\Rightarrow$ *real* $\Rightarrow$ $'c$ :: {*banach*, *second-countable-topology*}
  assumes *interval-lebesgue-integrable lborel a b g*
  assumes [*measurable*]: $\bigwedge y.\ y \in Y \Longrightarrow$ *set-borel-measurable borel* (*einterval* a b) (f y)
  assumes $\bigwedge y.\ y \in Y \Longrightarrow$ (*AE* t∈*einterval* a b *in lborel*. *norm* $(f\ y\ t) \le g\ t$)
  assumes $a < b$
  shows *uniform-limit* Y $(\lambda b'\ y.$ *LBINT* t=a..b'. $f\ y\ t)$ $(\lambda y.$ *LBINT* t=a..b. $f\ y\ t)$ (*at-left* b)
proof (*cases* $Y = \{\}$)
  case *False*
  have *g-nonneg*: *AE* t∈*einterval* a b *in lborel*. $g\ t \ge 0$
  proof −
    from ‹$Y \ne \{\}$› **obtain** y **where** $y \in Y$ **by** *auto*
    from *assms*(*3*)[*OF this*] **show** *?thesis*
      by *eventually-elim* (*auto elim*: *order.trans*[*rotated*])
  qed

  have *ev*: *eventually* $(\lambda b'.\ b' \in \{a{<}..{<}b\})$ (*at-left* b)
    using ‹$a < b$› **by** (*intro eventually-at-leftI*)

with ‹$a < b$› **have** *?thesis* $\longleftrightarrow$ *uniform-limit* Y $(\lambda b'\ y.\ \int t \in einterval\ a\ (min\ b\ b').\ f\ y\ t\ \partial lborel)$
$(\lambda y.\ \int t \in einterval\ a\ b.\ f\ y\ t\ \partial lborel)$ (*at-left* b)
by (*intro filterlim-cong arg-cong2*[**where** f = *uniformly-on*])
(*auto simp*: *interval-lebesgue-integral-def fun-eq-iff min-def*
intro!: *eventually-mono*[*OF ev*])
also have ...
proof (*rule uniform-limit-set-lebesgue-integral*[**where** $g = g$], *goal-cases*)
show $\forall_F\ b'$ *in at-left* b. *einterval* a (*min* b b') $\in$ *sets borel* $\wedge$
einterval a (*min* b b') $\subseteq$ *einterval* a b
using *ev* **by** *eventually-elim* (*auto simp*: *einterval-def*)
next
show (($\lambda b'$. *set-lebesgue-integral lborel* (*einterval* a (*min* b b')) g) $\longrightarrow$
set-lebesgue-integral lborel (*einterval* a b) g) (*at-left* b)
unfolding *set-lebesgue-integral-def*
proof (*intro tendsto-at-left-erealI-sequentially integral-dominated-convergence*)
have $*$: *set-borel-measurable borel* (*einterval* a b) g
using *assms*(*1*) *less-imp-le*[*OF* ‹$a < b$›]
by (*simp add*: *interval-lebesgue-integrable-def set-integrable-def set-borel-measurable-def*)
show (λx. *indicat-real* (*einterval* a b) $x *_R g\ x$) $\in$ *borel-measurable lborel*
using $*$ **by** (*simp add*: *set-borel-measurable-def*)
fix X :: *nat* $\Rightarrow$ *ereal* **and** n :: *nat*
have *set-borel-measurable borel* (*einterval* a (*min* b (X n))) g
by (*rule set-borel-measurable-subset*[*OF* $*$]) (*auto simp*: *einterval-def*)
thus (λx. *indicat-real* (*einterval* a (*min* b (X n))) $x *_R g\ x$) $\in$ *borel-measurable lborel*
by (*simp add*: *set-borel-measurable-def*)
next
fix X :: *nat* $\Rightarrow$ *ereal*
assume X: *filterlim* X (*at-left* b) *sequentially*
show *AE* x *in lborel*. (λn. *indicat-real* (*einterval* a (*min* b (X n))) $x *_R g\ x$)
$\longrightarrow$ *indicat-real* (*einterval* a b) $x *_R g\ x$
proof (*rule AE-I2*)
fix x :: *real*
have (λt. *indicator* (*einterval* a (*min* b (X t))) x :: *real*) $\longrightarrow$
indicator (*einterval* a b) x
proof (*cases* $x \in$ *einterval* a b)
case *False*
hence $x \notin$ *einterval* a (*min* b (X t))**for** t **by** (*auto simp*: *einterval-def*)
with *False* **show** *?thesis* **by** (*simp add*: *indicator-def*)
next
case *True*
with ‹$a < b$› **have** *eventually* (λt. $t \in \{max\ a\ x<..<b\}$) (*at-left* b)
by (*intro eventually-at-leftI*[*of ereal x*]) (*auto simp*: *einterval-def min-def*)
from *this* **and** X **have** *eventually* (λt. $X\ t \in \{max\ a\ x<..<b\}$) *sequentially*
by (*rule eventually-compose-filterlim*)
hence *eventually* (λt. *indicator* (*einterval* a (*min* b (X t))) x = (*1* :: *real*)) *sequentially*
by *eventually-elim* (*use True* **in** ‹*auto simp*: *indicator-def einterval-def*›)

from *tendsto-eventually*[*OF this*] **and** *True* **show** *?thesis*
by (*simp add*: *indicator-def*)
qed
thus ($\lambda n.$ *indicat-real* (*einterval a* (*min b* (*X n*))) $x *_R g\ x$)
$\longrightarrow$ *indicat-real* (*einterval a b*) $x *_R g\ x$ **by** (*intro tendsto-intros*)
qed
next
fix X :: *nat* $\Rightarrow$ *ereal* **and** n :: *nat*
show *AE x in lborel. norm* (*indicator* (*einterval a* (*min b* (*X n*))) $x *_R g\ x$) $\leq$
indicator (*einterval a b*) $x *_R g\ x$
using *g-nonneg* **by** *eventually-elim* (*auto simp*: *indicator-def einterval-def*)
qed (*use assms less-imp-le*[*OF* ‹$a < b$›] **in**
‹*auto simp*: *interval-lebesgue-integrable-def set-integrable-def*›)
qed (*use assms* **in** ‹*auto simp*: *interval-lebesgue-integrable-def*›)
finally show *?thesis* **.**
qed *auto*

lemma *uniform-limit-interval-integral-left*:
fixes f :: $'a \Rightarrow$ *real* $\Rightarrow$ $'c$ :: {*banach*, *second-countable-topology*}
assumes *interval-lebesgue-integrable lborel a b g*
assumes [*measurable*]: $\bigwedge y.\ y \in Y \Longrightarrow$ *set-borel-measurable borel* (*einterval a b*) (*f y*)
assumes $\bigwedge y.\ y \in Y \Longrightarrow$ (*AE t*$\in$*einterval a b in lborel. norm* (*f y t*) $\leq$ *g t*)
assumes $a < b$
shows *uniform-limit Y* ($\lambda a'\ y.$ *LBINT t=a'..b. f y t*) ($\lambda y.$ *LBINT t=a..b. f y t*) (*at-right a*)
proof (*cases* $Y = \{\}$)
case *False*
have *g-nonneg*: *AE t*$\in$*einterval a b in lborel.* $g\ t \geq 0$
proof $-$
from ‹$Y \neq \{\}$› **obtain** y **where** $y \in Y$ **by** *auto*
from *assms*(*3*)[*OF this*] **show** *?thesis*
by *eventually-elim* (*auto elim*: *order.trans*[*rotated*])
qed

have *ev*: *eventually* ($\lambda b'.\ b' \in \{a<..<b\}$) (*at-right a*)
using ‹$a < b$› **by** (*intro eventually-at-rightI*)
with ‹$a < b$› **have** *?thesis* $\longleftrightarrow$ *uniform-limit Y* ($\lambda a'\ y.$ $\int$ *t*$\in$*einterval* (*max a a'*) *b. f y t* ∂*lborel*)
($\lambda y.$ $\int$ *t*$\in$*einterval a b. f y t* ∂*lborel*) (*at-right a*)
by (*intro filterlim-cong arg-cong2*[**where** f = *uniformly-on*])
(*auto simp*: *interval-lebesgue-integral-def fun-eq-iff max-def*
intro!: *eventually-mono*[*OF ev*])
also have ...
proof (*rule uniform-limit-set-lebesgue-integral*[**where** $g = g$], *goal-cases*)
show $\forall_F$ a' *in at-right a. einterval* (*max a a'*) $b \in$ *sets borel* $\wedge$
einterval (*max a a'*) $b \subseteq$ *einterval a b*
using *ev* **by** *eventually-elim* (*auto simp*: *einterval-def*)

```
  next
    show ((λa'. set-lebesgue-integral lborel (einterval (max a a') b) g) ⟶
            set-lebesgue-integral lborel (einterval a b) g) (at-right a)
      unfolding set-lebesgue-integral-def
   proof (intro tendsto-at-right-erealI-sequentially integral-dominated-convergence)
      have *: set-borel-measurable borel (einterval a b) g
        using assms(1) less-imp-le[OF ‹a < b›]
      by (simp add: interval-lebesgue-integrable-def set-integrable-def set-borel-measurable-def)
      show (λx. indicat-real (einterval a b) x *R g x) ∈ borel-measurable lborel
        using * by (simp add: set-borel-measurable-def)
      fix X :: nat ⇒ ereal and n :: nat
      have set-borel-measurable borel (einterval (max a (X n)) b) g
        by (rule set-borel-measurable-subset[OF *]) (auto simp: einterval-def)
     thus (λx. indicat-real (einterval (max a (X n)) b) x *R g x) ∈ borel-measurable
lborel
        by (simp add: set-borel-measurable-def)
    next
      fix X :: nat ⇒ ereal
      assume X: filterlim X (at-right a) sequentially
      show AE x in lborel. (λn. indicat-real (einterval (max a (X n)) b) x *R g x)
              ⟶ indicat-real (einterval a b) x *R g x
      proof (rule AE-I2)
        fix x :: real
        have (λt. indicator (einterval (max a (X t)) b) x :: real) ⟶
                indicator (einterval a b) x
        proof (cases x ∈ einterval a b)
          case False
          hence x ∉ einterval (max a (X t)) bfor t by (auto simp: einterval-def)
          with False show ?thesis by (simp add: indicator-def)
        next
          case True
          with ‹a < b› have eventually (λt. t ∈ {a<..<x}) (at-right a)
               by (intro eventually-at-rightI[of - ereal x]) (auto simp: einterval-def
min-def)
          from this and X have eventually (λt. X t ∈ {a<..<x}) sequentially
            by (rule eventually-compose-filterlim)
         hence eventually (λt. indicator (einterval (max a (X t)) b) x = (1 :: real))
sequentially
           by eventually-elim (use True in ‹auto simp: indicator-def einterval-def›)
          from tendsto-eventually[OF this] and True show ?thesis
            by (simp add: indicator-def)
        qed
        thus (λn. indicat-real (einterval (max a (X n)) b) x *R g x)
               ⟶ indicat-real (einterval a b) x *R g x by (intro tendsto-intros)
      qed
    next
      fix X :: nat ⇒ ereal and n :: nat
      show AE x in lborel. norm (indicator (einterval (max a (X n)) b) x *R g x)
≤
```

indicator (*einterval a b*) *x* $*_R$ *g x*
using *g-nonneg* **by** *eventually-elim* (*auto simp*: *indicator-def einterval-def*)
qed (*use assms less-imp-le*[*OF* ‹*a* < *b*›] **in**
‹*auto simp*: *interval-lebesgue-integrable-def set-integrable-def*›)
qed (*use assms* **in** ‹*auto simp*: *interval-lebesgue-integrable-def*›)
finally show *?thesis* **.**
qed *auto*

lemma *uniform-limit-interval-integral-sequentially*:
fixes *f* :: *'a* ⇒ *real* ⇒ *'c* :: {*banach*, *second-countable-topology*}
assumes *interval-lebesgue-integrable lborel a b g*
assumes [*measurable*]: ⋀*y*. *y* ∈ *Y* ⟹ *set-borel-measurable borel* (*einterval a b*) (*f y*)
assumes ⋀*y*. *y* ∈ *Y* ⟹ (*AE t*∈*einterval a b in lborel*. *norm* (*f y t*) ≤ *g t*)
assumes *a'*: *filterlim a'* (*at-right a*) *sequentially*
assumes *b'*: *filterlim b'* (*at-left b*) *sequentially*
assumes *a* < *b*
shows *uniform-limit Y* (λ*n y*. *LBINT t*=*a' n*..*b' n*. *f y t*)
(λ*y*. *LBINT t*=*a*..*b*. *f y t*) *sequentially*
proof (*cases Y* = {})
case *False*
have *g-nonneg*: *AE t*∈*einterval a b in lborel*. *g t* ≥ *0*
proof −
from ‹*Y* ≠ {}› **obtain** *y* **where** *y* ∈ *Y* **by** *auto*
from *assms*(*3*)[*OF this*] **show** *?thesis*
by *eventually-elim* (*auto elim*: *order.trans*[*rotated*])
qed

have *ev*: *eventually* (λ*n*. *a* < *a' n* ∧ *a' n* < *b' n* ∧ *b' n* < *b*) *sequentially*
proof −
from *ereal-dense2*[*OF* ‹*a* < *b*›] **obtain** *t* **where** *t*: *a* < *ereal t ereal t* < *b* **by** *blast*
from *t* **have** *eventually* (λ*n*. *a' n* ∈ {*a*<..<*t*}) *sequentially*
by (*intro eventually-compose-filterlim*[*OF* - *a'*] *eventually-at-rightI*[*of* - *ereal t*])
moreover from *t* **have** *eventually* (λ*n*. *b' n* ∈ {*t*<..<*b*}) *sequentially*
by (*intro eventually-compose-filterlim*[*OF* - *b'*] *eventually-at-leftI*[*of ereal t*])
ultimately show *eventually* (λ*n*. *a* < *a' n* ∧ *a' n* < *b' n* ∧ *b' n* < *b*) *sequentially*
by *eventually-elim auto*
qed

have *?thesis* ⟷ *uniform-limit Y* (λ*n y*. ∫ *t*∈*einterval* (*max a* (*a' n*)) (*min b* (*b' n*)). *f y t* ∂*lborel*)
(λ*y*. ∫ *t*∈*einterval a b*. *f y t* ∂*lborel*) *sequentially* **using** ‹*a* < *b*›
by (*intro filterlim-cong arg-cong2*[**where** *f* = *uniformly-on*])
(*auto simp*: *interval-lebesgue-integral-def fun-eq-iff min-def max-def*
intro!: *eventually-mono*[*OF ev*])
also have ...
proof (*rule uniform-limit-set-lebesgue-integral*[**where** *g* = *g*], *goal-cases*)

show ∀$_F$ *n in sequentially. einterval* (*max a* (*a′ n*)) (*min b* (*b′ n*)) ∈ *sets borel* ∧
einterval (*max a* (*a′ n*)) (*min b* (*b′ n*)) ⊆ *einterval a b*
using *ev* **by** *eventually-elim* (*auto simp*: *einterval-def*)
next
show ((λ*n. set-lebesgue-integral lborel* (*einterval* (*max a* (*a′ n*)) (*min b* (*b′ n*))) *g*) ⟶
set-lebesgue-integral lborel (*einterval a b*) *g*) *sequentially*
unfolding *set-lebesgue-integral-def*
proof (*intro integral-dominated-convergence*)
have ∗: *set-borel-measurable borel* (*einterval a b*) *g*
using *assms*(*1*) *less-imp-le*[*OF* ‹*a* < *b*›]
by (*simp add*: *interval-lebesgue-integrable-def set-integrable-def set-borel-measurable-def*)
show (λ*x. indicat-real* (*einterval a b*) *x* ∗$_R$ *g x*) ∈ *borel-measurable lborel*
using ∗ **by** (*simp add*: *set-borel-measurable-def*)
fix *n* :: *nat*
have *set-borel-measurable borel* (*einterval* (*max a* (*a′ n*)) (*min b* (*b′ n*))) *g*
by (*rule set-borel-measurable-subset*[*OF* ∗]) (*auto simp*: *einterval-def*)
thus (λ*x. indicat-real* (*einterval* (*max a* (*a′ n*)) (*min b* (*b′ n*))) *x* ∗$_R$ *g x*) ∈ *borel-measurable lborel*
by (*simp add*: *set-borel-measurable-def*)
next
show *AE x in lborel.* (λ*n. indicat-real* (*einterval* (*max a* (*a′ n*)) (*min b* (*b′ n*))) *x* ∗$_R$ *g x*)
⟶ *indicat-real* (*einterval a b*) *x* ∗$_R$ *g x*
proof (*rule AE-I2*)
fix *x* :: *real*
have (λ*t. indicator* (*einterval* (*max a* (*a′ t*)) (*min b* (*b′ t*))) *x* :: *real*) ⟶
indicator (*einterval a b*) *x*
proof (*cases x* ∈ *einterval a b*)
case *False*
hence *x* ∉ *einterval* (*max a* (*a′ t*)) (*min b* (*b′ t*))**for** *t*
by (*auto simp*: *einterval-def*)
with *False* **show** *?thesis* **by** (*simp add*: *indicator-def*)
next
case *True*
with ‹*a* < *b*› **have** *eventually* (λ*t. t* ∈ {*a*<..<*x*}) (*at-right a*)
by (*intro eventually-at-rightI*[*of - ereal x*]) (*auto simp*: *einterval-def min-def*)

have *eventually* (λ*n. x* ∈ {*a′ n*<..<*b′ n*}) *sequentially*
proof −
have *eventually* (λ*n. a′ n* ∈ {*a*<..<*x*}) *sequentially* **using** *True*
by (*intro eventually-compose-filterlim*[*OF - a′*] *eventually-at-rightI*[*of - ereal x*])
(*auto simp*: *einterval-def*)
moreover have *eventually* (λ*n. b′ n* ∈ {*x*<..<*b*}) *sequentially* **using** *True*
by (*intro eventually-compose-filterlim*[*OF - b′*] *eventually-at-leftI*[*of*

ereal x])

(*auto simp*: *einterval-def*)

ultimately show *eventually* ($\lambda n.$ $x \in \{a'\ n{<}..{<}b'\ n\}$) *sequentially*

by *eventually-elim auto*

qed

hence *eventually* ($\lambda t.$ *indicator* (*einterval* (*max a* (a' *t*)) (*min b* (b' *t*))) *x* = (*1* :: *real*)) *sequentially*

by *eventually-elim* (*use True* **in** ‹*auto simp*: *indicator-def einterval-def*›)

from *tendsto-eventually*[*OF this*] **and** *True* **show** *?thesis*

by (*simp add*: *indicator-def*)

qed

thus ($\lambda n.$ *indicat-real* (*einterval* (*max a* (a' *n*)) (*min b* (b' *n*))) $x *_R g\ x$)

$\longrightarrow$ *indicat-real* (*einterval a b*) $x *_R g\ x$ **by** (*intro tendsto-intros*)

qed

next

fix *X* :: *nat* $\Rightarrow$ *ereal* **and** *n* :: *nat*

show *AE x in lborel. norm* (*indicator* (*einterval* (*max a* (a' *n*)) (*min b* (b' *n*))) $x *_R g\ x$) $\leq$

indicator (*einterval a b*) $x *_R g\ x$

using *g-nonneg* **by** *eventually-elim* (*auto simp*: *indicator-def einterval-def*)

qed (*use assms less-imp-le*[*OF* ‹$a < b$›] **in**

‹*auto simp*: *interval-lebesgue-integrable-def set-integrable-def*›)

qed (*use assms* **in** ‹*auto simp*: *interval-lebesgue-integrable-def*›)

finally show *?thesis* **.**

qed *auto*

lemma *interval-lebesgue-integrable-combine*:

assumes *interval-lebesgue-integrable lborel A B f*

assumes *interval-lebesgue-integrable lborel B C f*

assumes *set-borel-measurable borel* (*einterval A C*) *f*

assumes $A \leq B$ $B \leq C$

shows *interval-lebesgue-integrable lborel A C f*

proof –

have *meas*: *set-borel-measurable borel* (*einterval A B* $\cup$ *einterval B C*) *f*

by (*rule set-borel-measurable-subset*[*OF assms*(*3*)]) (*use assms* **in** ‹*auto simp*: *einterval-def*›)

have *set-integrable lborel* (*einterval A B* $\cup$ *einterval B C*) *f*

using *assms* **by** (*intro set-integrable-Un*) (*auto simp*: *interval-lebesgue-integrable-def*)

also have *?this* $\longleftrightarrow$ *set-integrable lborel* (*einterval A C*) *f*

proof (*cases* $B \in \{\infty, -\infty\}$)

case *True*

with *assms* **have** *einterval A B* $\cup$ *einterval B C* = *einterval A C*

by (*auto simp*: *einterval-def*)

thus *?thesis* **by** *simp*

next

case *False*

then obtain B' **where** [*simp*]: B = *ereal* B'

by (*cases B*) *auto*

have *indicator* (*einterval A C*) *x* = (*indicator* (*einterval A B* $\cup$ *einterval B C*)

x :: *real*)
    if $x \neq B'$ **for** x **using** *assms*(*4*,*5*) *that*
    by (*cases A*; *cases C*) (*auto simp*: *einterval-def indicator-def*)
  hence $\{x \in$ *space lborel*. *indicat-real* (*einterval A B* $\cup$ *einterval B C*) $x *_R f\,x$ $\neq$
      indicat-real (*einterval A C*) $x *_R f\,x\} \subseteq \{B'\}$ **by** *force*
  thus *?thesis* **unfolding** *set-integrable-def* **using** *meas assms*
    by (*intro integrable-cong-AE AE-I*[*of* - - $\{B'\}$])
      (*simp-all add*: *set-borel-measurable-def*)
 qed
 also have ... $\longleftrightarrow$ *?thesis*
  using *order.trans*[*OF assms*(*4*,*5*)] **by** (*simp add*: *interval-lebesgue-integrable-def*)
 finally show *?thesis* **.**
qed

lemma *interval-lebesgue-integrable-bigo-right*:
 fixes *A B* :: *real*
 fixes *f* :: *real* $\Rightarrow$ *real*
 assumes $f \in O$[*at-left B*](g)
 assumes *cont*: *continuous-on* {*A*..<*B*} *f*
 assumes *meas*: *set-borel-measurable borel* {*A*<..<*B*} *f*
 assumes *interval-lebesgue-integrable lborel A B g*
 assumes $A < B$
 shows *interval-lebesgue-integrable lborel A B f*
proof −
 from *assms*(*1*) **obtain** *c* **where** *c*: $c > 0$ *eventually* (λx. *norm* ($f\,x$) $\leq c *$ *norm* ($g\,x$)) (*at-left B*)
  by (*elim landau-o.bigE*)
 then obtain B' **where** B': $B' < B$ $\bigwedge x.\ x \in \{B'<..<B\} \Longrightarrow$ *norm* ($f\,x$) $\leq c *$ *norm* ($g\,x$)
  using ‹$A < B$› **by** (*auto simp*: *Topological-Spaces.eventually-at-left*[*of A*])

 show *?thesis*
 proof (*rule interval-lebesgue-integrable-combine*)
  show *interval-lebesgue-integrable lborel A* (*max A B'*) *f*
    using B' *assms*
      by (*intro interval-integrable-continuous-on continuous-on-subset*[*OF cont*]) *auto*
  show *set-borel-measurable borel* (*einterval* (*ereal A*) (*ereal B*)) *f*
    using *assms* **by** *simp*
  have *meas'*: *set-borel-measurable borel* {*max A B'*<..<*B*} *f*
    by (*rule set-borel-measurable-subset*[*OF meas*]) *auto*
  have *set-integrable lborel* {*max A B'*<..<*B*} *f*
  proof (*rule set-integrable-bound*[*OF* - - *AE-I2*[*OF impI*]])
    have *set-integrable lborel* {*A*<..<*B*} ($\lambda x.\ c * g\ x$)
      using *assms* **by** (*simp add*: *interval-lebesgue-integrable-def*)
    thus *set-integrable lborel* {*max A B'*<..<*B*} ($\lambda x.\ c * g\ x$)
      by (*rule set-integrable-subset*) *auto*
  next

```
      fix x assume x ∈ {max A B′<..<B}
      hence norm (f x) ≤ c ∗ norm (g x)
        by (intro B′) auto
      also have . . . ≤ norm (c ∗ g x)
        unfolding norm-mult by (intro mult-right-mono) auto
      finally show  norm (f x) ≤ norm (c ∗ g x) .
    qed (use meas′ in ‹simp-all add: set-borel-measurable-def›)
    thus interval-lebesgue-integrable lborel (ereal (max A B′)) (ereal B) f
       unfolding interval-lebesgue-integrable-def einterval-eq-Icc using ‹B′ < B›
assms by simp
  qed (use B′ assms in auto)
qed

lemma interval-lebesgue-integrable-bigo-left:
  fixes A B :: real
  fixes f :: real ⇒ real
  assumes f ∈ O[at-right A](g)
  assumes cont: continuous-on {A<..B} f
  assumes meas: set-borel-measurable borel {A<..<B} f
  assumes interval-lebesgue-integrable lborel A B g
  assumes A < B
  shows   interval-lebesgue-integrable lborel A B f
proof −
  from assms(1) obtain c where c: c > 0 eventually (λx. norm (f x) ≤ c ∗ norm
(g x)) (at-right A)
    by (elim landau-o.bigE)
  then obtain A′ where A′: A′ > A ⋀x. x ∈ {A<..<A′} ⟹ norm (f x) ≤ c ∗
norm (g x)
    using ‹A < B› by (auto simp: Topological-Spaces.eventually-at-right[of A])

  show ?thesis
  proof (rule interval-lebesgue-integrable-combine)
    show interval-lebesgue-integrable lborel (min B A′) B f
      using A′ assms
        by (intro interval-integrable-continuous-on continuous-on-subset[OF cont])
auto
    show set-borel-measurable borel (einterval (ereal A) (ereal B)) f
      using assms by simp
    have meas′: set-borel-measurable borel {A<..<min B A′} f
      by (rule set-borel-measurable-subset[OF meas]) auto
    have set-integrable lborel {A<..<min B A′} f
    proof (rule set-integrable-bound[OF - - AE-I2[OF impI]])
      have set-integrable lborel {A<..<B} (λx. c ∗ g x)
        using assms by (simp add: interval-lebesgue-integrable-def)
      thus set-integrable lborel {A<..<min B A′} (λx. c ∗ g x)
        by (rule set-integrable-subset) auto
    next
      fix x assume x ∈ {A<..<min B A′}
      hence norm (f x) ≤ c ∗ norm (g x)
```

by (*intro A'*) *auto*
also have ... $\leq$ *norm* (*c* $*$ *g x*)
unfolding *norm-mult* **by** (*intro mult-right-mono*) *auto*
finally show *norm* (*f x*) $\leq$ *norm* (*c* $*$ *g x*) **.**
qed (*use meas'* **in** ‹*simp-all add*: *set-borel-measurable-def*›)
thus *interval-lebesgue-integrable lborel* (*ereal A*) (*ereal* (*min B A'*)) *f*
unfolding *interval-lebesgue-integrable-def einterval-eq-Icc* **using** ‹*A'* > *A*›
assms **by** *simp*
qed (*use A' assms* **in** *auto*)
qed

1.4 Other material

lemma *summable-comparison-test-bigo*:
fixes *f* :: *nat* ⇒ *real*
assumes *summable* (λ*n*. *norm* (*g n*)) *f* ∈ *O*(*g*)
shows *summable f*
proof −
from ‹*f* ∈ *O*(*g*)› **obtain** *C* **where** *C*: *eventually* (λ*x*. *norm* (*f x*) $\leq$ *C* $*$ *norm* (*g x*)) *at-top*
by (*auto elim*: *landau-o.bigE*)
thus *?thesis*
by (*rule summable-comparison-test-ev*) (*insert assms*, *auto intro*: *summable-mult*)
qed

lemma *fps-expansion-cong*:
assumes *eventually* (λ*x*. *g x* = *h x*) (*nhds x*)
shows *fps-expansion g x* = *fps-expansion h x*
proof −
have (*deriv* ^^ *n*) *g x* = (*deriv* ^^ *n*) *h x* **for** *n*
by (*intro higher-deriv-cong-ev assms refl*)
thus *?thesis* **by** (*simp add*: *fps-expansion-def*)
qed

lemma *fps-expansion-eq-zero-iff*:
assumes *g holomorphic-on ball z r r* > *0*
shows *fps-expansion g z* = *0* ⟷ (∀ *z*∈*ball z r*. *g z* = *0*)
proof
assume $*$: ∀ *z*∈*ball z r*. *g z* = *0*
have *eventually* (λ*w*. *w* ∈ *ball z r*) (*nhds z*)
using *assms* **by** (*intro eventually-nhds-in-open*) *auto*
hence *eventually* (λ*z*. *g z* = *0*) (*nhds z*)
by *eventually-elim* (*use* $*$ **in** *auto*)
hence *fps-expansion g z* = *fps-expansion* (λ-. *0*) *z*
by (*intro fps-expansion-cong*)
thus *fps-expansion g z* = *0*
by (*simp add*: *fps-expansion-def fps-zero-def*)
next
assume $*$: *fps-expansion g z* = *0*

have *g w* = *0* **if** *w* ∈ *ball z r* **for** *w*
by (*rule holomorphic-fun-eq-0-on-ball*[*OF assms*(*1*) *that*])
(*use* ∗ **in** ‹*auto simp*: *fps-expansion-def fps-eq-iff*›)
thus ∀ *w*∈*ball z r*. *g w* = *0* **by** *blast*
qed

lemma *fds-nth-higher-deriv*:
fds-nth ((*fds-deriv* ^^ *k*) *F*) = (λ*n*. (−*1*) ^ *k* ∗ *of-real* (*ln n*) ^ *k* ∗ *fds-nth F n*)
by (*induction k*) (*auto simp*: *fds-nth-deriv fun-eq-iff simp flip*: *scaleR-conv-of-real*)

lemma *binomial-n-n-minus-one* [*simp*]: *n* > *0* ⟹ *n choose* (*n* − *Suc 0*) = *n*
by (*cases n*) *auto*

lemma *has-field-derivative-complex-powr-right*:
w ≠ *0* ⟹ ((λ*z*. *w powr z*) *has-field-derivative Ln w* ∗ *w powr z*) (*at z within A*)
by (*rule DERIV-subset*, *rule has-field-derivative-powr-right*) *auto*

lemmas *has-field-derivative-complex-powr-right′* =
has-field-derivative-complex-powr-right[*THEN DERIV-chain2*]

end

2 The Hurwitz and Riemann ζ functions

theory *Zeta-Function*
imports
Euler-MacLaurin.Euler-MacLaurin
Bernoulli.Bernoulli-Zeta
Dirichlet-Series.Dirichlet-Series-Analysis
Winding-Number-Eval.Winding-Number-Eval
HOL−Real-Asymp.Real-Asymp
Zeta-Library
Pure−ex.Guess
begin

2.1 Preliminary facts

lemma *holomorphic-on-extend*:
assumes *f holomorphic-on S* − {ξ} ξ ∈ *interior S f* ∈ *O*[*at* ξ](λ-. *1*)
shows (∃ *g*. *g holomorphic-on S* ∧ (∀ *z*∈*S* − {ξ}. *g z* = *f z*))
by (*subst holomorphic-on-extend-bounded*) (*insert assms*, *auto elim*!: *landau-o.bigE*)

lemma *removable-singularities*:
assumes *finite X X* ⊆ *interior S f holomorphic-on* (*S* − *X*)
assumes ⋀*z*. *z* ∈ *X* ⟹ *f* ∈ *O*[*at z*](λ-. *1*)
shows ∃ *g*. *g holomorphic-on S* ∧ (∀ *z*∈*S*−*X*. *g z* = *f z*)
using *assms*
proof (*induction arbitrary*: *f rule*: *finite-induct*)
case *empty*

thus *?case* **by** *auto*
next
case (*insert z0 X f*)
from *insert.prems* **and** *insert.hyps* **have** *z0*: *z0* ∈ *interior* (*S* − *X*)
by (*auto simp*: *interior-diff finite-imp-closed*)
hence ∃ *g*. *g holomorphic-on* (*S* − *X*) ∧ (∀ *z*∈*S* − *X* − {*z0*}. *g z* = *f z*)
using *insert.prems insert.hyps* **by** (*intro holomorphic-on-extend*) *auto*
then obtain *g* **where** *g*: *g holomorphic-on* (*S* − *X*) ∀ *z*∈*S* − *X* − {*z0*}. *g z* = *f z* **by** *blast*
have ∃ *h*. *h holomorphic-on S* ∧ (∀ *z*∈*S* − *X*. *h z* = *g z*)
proof (*rule insert.IH*)
fix *z0'* **assume** *z0'*: *z0'* ∈ *X*
hence *eventually* (λ*z*. *z* ∈ *interior S* − (*X* − {*z0'*}) − {*z0*}) (*nhds z0'*)
using *insert.prems insert.hyps*
by (*intro eventually-nhds-in-open open-Diff finite-imp-closed*) *auto*
hence *ev*: *eventually* (λ*z*. *z* ∈ *S* − *X* − {*z0*}) (*at z0'*)
unfolding *eventually-at-filter*
by *eventually-elim* (*insert z0' insert.hyps interior-subset*[*of S*], *auto*)
have *g* ∈ Θ[*at z0'*](*f*)
by (*intro bigthetaI-cong eventually-mono*[*OF ev*]) (*insert g*, *auto*)
also have *f* ∈ *O*[*at z0'*](λ-. *1*)
using *z0'* **by** (*intro insert.prems*) *auto*
finally show *g* ∈ . . . **.**
qed (*insert insert.prems g*, *auto*)
then obtain *h* **where** *h holomorphic-on S* ∀ *z*∈*S* − *X*. *h z* = *g z* **by** *blast*
with *g* **have** *h holomorphic-on S* ∀ *z*∈*S* − *insert z0 X*. *h z* = *f z* **by** *auto*
thus *?case* **by** *blast*
qed

lemma *continuous-imp-bigo-1*:
assumes *continuous* (*at x within A*) *f*
shows *f* ∈ *O*[*at x within A*](λ-. *1*)
proof (*rule bigoI-tendsto*)
from *assms* **show** ((λ*x*. *f x* / *1*) ⟶ *f x*) (*at x within A*)
by (*auto simp*: *continuous-within*)
qed *auto*

lemma *taylor-bigo-linear*:
assumes *f field-differentiable at x0 within A*
shows (λ*x*. *f x* − *f x0*) ∈ *O*[*at x0 within A*](λ*x*. *x* − *x0*)
proof −
from *assms* **obtain** *f'* **where** (*f has-field-derivative f'*) (*at x0 within A*)
by (*auto simp*: *field-differentiable-def*)
hence ((λ*x*. (*f x* − *f x0*) / (*x* − *x0*)) ⟶ *f'*) (*at x0 within A*)
by (*auto simp*: *has-field-derivative-iff*)
thus *?thesis* **by** (*intro bigoI-tendsto*[**where** *c* = *f'*]) (*auto simp*: *eventually-at-filter*)
qed

lemma *powr-add-minus-powr-asymptotics*:
  fixes *a z* :: *complex*
  shows $((\lambda z.\ ((1 + z)\ powr\ a - 1)\ /\ z) \longrightarrow a)\ (at\ 0)$
proof (*rule Lim-transform-eventually*)
  have *eventually* $(\lambda z{::}complex.\ z \in ball\ 0\ 1 - \{0\})\ (at\ 0)$
    using *eventually-at-ball'*[*of 1 0::complex UNIV*] **by** (*simp add*: *dist-norm*)
  thus *eventually* $(\lambda z.\ (\sum n.\ (a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ n) = ((1 + z)\ powr\ a - 1)\ /\ z)\ (at\ 0)$
  proof *eventually-elim*
    case (*elim z*)
    hence $(\lambda n.\ (a\ gchoose\ n) * z\ \widehat{}\ n)\ sums\ (1 + z)\ powr\ a$
      by (*intro gen-binomial-complex*) *auto*
    hence $(\lambda n.\ (a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ (Suc\ n))\ sums\ ((1 + z)\ powr\ a - 1)$
      by (*subst sums-Suc-iff*) *simp-all*
    also have $(\lambda n.\ (a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ (Suc\ n)) = (\lambda n.\ z * ((a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ n))$
      by (*simp add*: *algebra-simps*)
    finally have $(\lambda n.\ (a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ n)\ sums\ (((1 + z)\ powr\ a - 1)\ /\ z)$
      by (*rule sums-mult-D*) (*use elim* **in** *auto*)
    thus *?case* **by** (*simp add*: *sums-iff*)
  qed
next
  have $conv\text{-}radius\ (\lambda n.\ a\ gchoose\ (n + 1)) = conv\text{-}radius\ (\lambda n.\ a\ gchoose\ n)$
    using *conv-radius-shift*[*of* $\lambda n.\ a\ gchoose\ n\ 1$] **by** *simp*
  hence *continuous-on* $(cball\ 0\ (1/2))\ (\lambda z.\ \sum n.\ (a\ gchoose\ (Suc\ n)) * (z - 0)\ \widehat{}\ n)$
    using *conv-radius-gchoose*[*of a*] **by** (*intro powser-continuous-suminf*) (*simp-all*)
  hence *isCont* $(\lambda z.\ \sum n.\ (a\ gchoose\ (Suc\ n)) * z\ \widehat{}\ n)\ 0$
    by (*auto intro*: *continuous-on-interior*)
  thus $(\lambda z.\ \sum n.\ (a\ gchoose\ Suc\ n) * z\ \widehat{}\ n)\ {-0\rightarrow}\ a$
    by (*auto simp*: *isCont-def*)
qed

lemma *complex-powr-add-minus-powr-asymptotics*:
  fixes *s* :: *complex*
  assumes *a*: $a > 0$ **and** *s*: $Re\ s < 1$
  shows *filterlim* $(\lambda x.\ of\text{-}real\ (x + a)\ powr\ s - of\text{-}real\ x\ powr\ s)\ (nhds\ 0)\ at\text{-}top$
proof (*rule Lim-transform-eventually*)
  show *eventually* $(\lambda x.\ ((1 + of\text{-}real\ (a\ /\ x))\ powr\ s - 1)\ /\ of\text{-}real\ (a\ /\ x) *$
      $of\text{-}real\ x\ powr\ (s - 1) * a =$
      $of\text{-}real\ (x + a)\ powr\ s - of\text{-}real\ x\ powr\ s)\ at\text{-}top$
    (**is** *eventually* $(\lambda x.\ ?f\ x\ /\ ?g\ x * ?h\ x * \text{-} = \text{-})\ \text{-}$) **using** *eventually-gt-at-top*[*of a*]
  proof *eventually-elim*
    case (*elim x*)
    have $?f\ x\ /\ ?g\ x * ?h\ x * a = ?f\ x * (a * ?h\ x\ /\ ?g\ x)$ **by** *simp*
    also have $a * ?h\ x\ /\ ?g\ x = of\text{-}real\ x\ powr\ s$

using *elim a* **by** (*simp add*: *powr-diff*)
also have *?f x* ∗ ... = *of-real* (*x* + *a*) *powr s* − *of-real x powr s*
using *a elim* **by** (*simp add*: *algebra-simps powr-times-real* [*symmetric*])
finally show *?case* **.**
qed

have *filterlim* (*λx. complex-of-real* (*a* / *x*)) (*nhds* (*complex-of-real 0*)) *at-top*
by (*intro tendsto-of-real real-tendsto-divide-at-top*[*OF tendsto-const*] *filterlim-ident*)
hence *filterlim* (*λx. complex-of-real* (*a* / *x*)) (*at 0*) *at-top*
using *a* **by** (*intro filterlim-atI*) *auto*
hence ((*λx. ?f x* / *?g x* ∗ *?h x* ∗ *a*) ⟶ *s* ∗ *0* ∗ *a*) *at-top* **using** *s*
by (*intro tendsto-mult filterlim-compose*[*OF powr-add-minus-powr-asymptotics*]
tendsto-const tendsto-neg-powr-complex-of-real filterlim-ident) *auto*
thus ((*λx. ?f x* / *?g x* ∗ *?h x* ∗ *a*) ⟶ *0*) *at-top* **by** *simp*
qed

lemma *summable-zeta*:
assumes *Re s* > *1*
shows *summable* (*λn. of-nat* (*Suc n*) *powr* −*s*)
proof −
have *summable* (*λn. exp* (*complex-of-real* (*ln* (*real* (*Suc n*))) ∗ − *s*)) (**is** *summable ?f*)
by (*subst summable-Suc-iff*, *rule summable-complex-powr-iff*) (*use assms* **in** *auto*)
also have *?f* = (*λn. of-nat* (*Suc n*) *powr* −*s*)
by (*simp add*: *powr-def algebra-simps del*: *of-nat-Suc*)
finally show *?thesis* **.**
qed

lemma *summable-zeta-real*:
assumes *x* > *1*
shows *summable* (*λn. real* (*Suc n*) *powr* −*x*)
proof −
have *summable* (*λn. of-nat* (*Suc n*) *powr* −*complex-of-real x*)
using *assms* **by** (*intro summable-zeta*) *simp-all*
also have (*λn. of-nat* (*Suc n*) *powr* −*complex-of-real x*) = (*λn. of-real* (*real* (*Suc n*) *powr* −*x*))
by (*subst powr-Reals-eq*) *simp-all*
finally show *?thesis*
by (*subst* (*asm*) *summable-complex-of-real*)
qed

lemma *summable-hurwitz-zeta*:
assumes *Re s* > *1 a* > *0*
shows *summable* (*λn.* (*of-nat n* + *of-real a*) *powr* −*s*)
proof −
have *summable* (*λn.* (*of-nat* (*Suc n*) + *of-real a*) *powr* −*s*)
proof (*rule summable-comparison-test′* [*OF summable-zeta-real* [*OF assms*(*1*)]]

)
  fix *n* :: *nat*
  have *norm* ((*of-nat* (*Suc n*) + *of-real a*) *powr* −*s*) = (*real* (*Suc n*) + *a*) *powr* − *Re s*
    (**is** *?N* = -) **using** *assms* **by** (*simp add*: *norm-powr-real-powr*)
  also have . . . ≤ *real* (*Suc n*) *powr* −*Re s*
    using *assms* **by** (*intro powr-mono2'*) *auto*
  finally show *?N* ≤ . . . **.**
 qed
 thus *?thesis* **by** (*subst* (*asm*) *summable-Suc-iff*)
qed

lemma *summable-hurwitz-zeta-real*:
 assumes *x* > *1* *a* > *0*
 shows *summable* (λ*n*. (*real n* + *a*) *powr* −*x*)
proof −
 have *summable* (λ*n*. (*of-nat n* + *of-real a*) *powr* −*complex-of-real x*)
  using *assms* **by** (*intro summable-hurwitz-zeta*) *simp-all*
 also have (λ*n*. (*of-nat n* + *of-real a*) *powr* −*complex-of-real x*) =
    (λ*n*. *of-real* ((*real n* + *a*) *powr* −*x*))
  using *assms* **by** (*subst powr-Reals-eq*) *simp-all*
 finally show *?thesis*
  by (*subst* (*asm*) *summable-complex-of-real*)
qed

2.2 Definitions

We use the Euler–MacLaurin summation formula to express $\zeta(s,a) - \frac{a^{1-s}}{s-1}$ as a polynomial plus some remainder term, which is an integral over a function of order $O(-1-2n-\Re(s))$. It is then clear that this integral converges uniformly to an analytic function in s for all s with $\Re(s) > -2n$.

definition *pre-zeta-aux* :: *nat* ⇒ *real* ⇒ *complex* ⇒ *complex* **where**
 pre-zeta-aux N a s = *a powr* − *s* / *2* +
  (∑ *i*=*1*..*N*. (*bernoulli* (*2* ∗ *i*) / *fact* (*2* ∗ *i*)) $*_R$ (*pochhammer s* (*2*∗*i* − *1*) ∗
      of-real a powr (− *s* − *of-nat* (*2*∗*i* − *1*)))) +
  EM-remainder (*Suc* (*2*∗*N*))
   (λ*x*. −(*pochhammer s* (*Suc* (*2*∗*N*)) ∗ *of-real* (*x* + *a*) *powr* (− *1* − *2*∗*N* − *s*))) *0*

By iterating the above construction long enough, we can extend this to the entire complex plane.

definition *pre-zeta* :: *real* ⇒ *complex* ⇒ *complex* **where**
 pre-zeta a s = *pre-zeta-aux* (*nat* (*1* − ⌈*Re s* / *2*⌉)) *a s*

We can then obtain the Hurwitz ζ function by adding back the pole at 1. Note that it is not necessary to trust that this somewhat complicated definition is, in fact, the correct one, since we will later show that this

Hurwitz zeta function fulfils

$$\zeta(s,a) = \sum_{n=0}^{\infty} \frac{1}{(n+a)^s}$$

and is analytic on $\mathbb{C}\setminus\{1\}$, which uniquely defines the function due to analytic continuation. It is therefore obvious that any alternative definition that is analytic on $\mathbb{C} \setminus \{1\}$ and satisfies the above equation must be equal to our Hurwitz ζ function.

definition *hurwitz-zeta* :: *real* ⇒ *complex* ⇒ *complex* **where**
hurwitz-zeta a s = (*if s* = *1 then 0 else pre-zeta a s* + *of-real a powr* (*1* − *s*) / (*s* − *1*))

The Riemann ζ function is simply the Hurwitz ζ function with $a = 1$.

definition *zeta* :: *complex* ⇒ *complex* **where**
zeta = *hurwitz-zeta 1*

We define the ζ functions as 0 at their poles. To avoid confusion, these facts are not added as simplification rules by default.

lemma *hurwitz-zeta-1*: *hurwitz-zeta c 1* = *0*
by (*simp add*: *hurwitz-zeta-def*)

lemma *zeta-1*: *zeta 1* = *0*
by (*simp add*: *zeta-def hurwitz-zeta-1*)

lemma *zeta-minus-pole-eq*: *s* ≠ *1* ⟹ *zeta s* − *1* / (*s* − *1*) = *pre-zeta 1 s*
by (*simp add*: *zeta-def hurwitz-zeta-def*)

context
begin

private lemma *holomorphic-pre-zeta-aux'*:
assumes *a* > *0 bounded U open U U* ⊆ {*s*. *Re s* > *σ*} **and** *σ*: *σ* > − *2* ∗ *real n*
shows *pre-zeta-aux n a holomorphic-on U* **unfolding** *pre-zeta-aux-def*
proof (*intro holomorphic-intros*)
define *C* :: *real* **where** *C* = *max 0* (*Sup* ((λ*s*. *norm* (*pochhammer s* (*Suc* (*2* ∗ *n*)))) ‘ *closure U*))
have *compact* (*closure U*)
using *assms* **by** (*auto simp*: *compact-eq-bounded-closed*)
hence *compact* ((λ*s*. *norm* (*pochhammer s* (*Suc* (*2* ∗ *n*)))) ‘ *closure U*)
by (*rule compact-continuous-image* [*rotated*]) (*auto intro*!: *continuous-intros*)
hence *bounded* ((λ*s*. *norm* (*pochhammer s* (*Suc* (*2* ∗ *n*)))) ‘ *closure U*)
by (*simp add*: *compact-eq-bounded-closed*)
hence *C*: *cmod* (*pochhammer s* (*Suc* (*2* ∗ *n*))) ≤ *C* **if** *s* ∈ *U* **for** *s*
using *that closure-subset*[*of U*] **unfolding** *C-def*
by (*intro max.coboundedI2 cSup-upper bounded-imp-bdd-above*) (*auto simp*: *image-iff*)

have C' [*simp*]: $C \geq 0$ **by** (*simp add*: *C-def*)

let *?g* = λ(*x*::*real*). $C * (x + a)$ *powr* ($-$ *1* $-$ *2* $*$ *of-nat n* $- \sigma$)
let *?G* = λ(*x*::*real*). C / ($-$ *2* $*$ *of-nat n* $- \sigma$) $* (x + a)$ *powr* ($-$ *2* $*$ *of-nat n* $- \sigma$)
define *poch'* **where** *poch'* = *deriv* (λ*z*::*complex*. *pochhammer z* (*Suc* (*2* $*$ *n*)))
have [*derivative-intros*]:
((*λz*. *pochhammer z* (*Suc* (*2* $*$ *n*))) *has-field-derivative poch' z*) (*at z within A*)
for *z* :: *complex* **and** *A* **unfolding** *poch'-def*
by (*rule holomorphic-derivI* [*OF holomorphic-pochhammer* [*of - UNIV*]]) *auto*
have *A*: *continuous-on A poch'* **for** *A* **unfolding** *poch'-def*
by (*rule continuous-on-subset*[*OF - subset-UNIV*],
intro holomorphic-on-imp-continuous-on holomorphic-deriv)
(*auto intro*: *holomorphic-pochhammer*)
note [*continuous-intros*] = *continuous-on-compose2*[*OF this - subset-UNIV*]

define *f'* **where** *f'* = (*λz t*. $-$ (*poch' z* $*$ *complex-of-real* (*t* + *a*) *powr* ($-$ *1* $-$ *2* $*$ *of-nat n* $-$ *z*) $-$
Ln (*complex-of-real* (*t* + *a*)) $*$ *complex-of-real* (*t* + *a*) *powr*
($-$ *1* $-$ *2* $*$ *of-nat n* $-$ *z*) $*$ *pochhammer z* (*Suc* (*2* $*$ *n*))))

show (*λz*. *EM-remainder* (*Suc* (*2* $*$ *n*)) (*λx*. $-$ (*pochhammer z* (*Suc* (*2* $*$ *n*)) $*$
complex-of-real (*x* + *a*) *powr* ($-$ *1* $-$ *2* $*$ *of-nat n* $-$ *z*))) *0*) *holomorphic-on*
U **unfolding** *pre-zeta-aux-def*
proof (*rule holomorphic-EM-remainder*[*of - ?G ?g - - f'*], *goal-cases*)
case (*1 x*)
show *?case*
by (*insert 1* σ ‹*a* > *0*›, *rule derivative-eq-intros refl* | *simp*)+
(*auto simp*: *field-simps powr-diff powr-add powr-minus*)
next
case (*2 z t x*)
note [*derivative-intros*] = *has-field-derivative-powr-right* [*THEN DERIV-chain2*]
show *?case*
by (*insert 2* σ ‹*a* > *0*›, (*rule derivative-eq-intros refl* | (*simp add*: *add-eq-0-iff*; *fail*))+)
(*simp add*: *f'-def*)
next
case *3*
hence $*$: *complex-of-real x* + *complex-of-real a* $\notin \mathbb{R}_{\leq 0}$ **if** $x \geq 0$ **for** *x*
using *nonpos-Reals-of-real-iff*[*of x+a*, *unfolded of-real-add*] *that* ‹*a* > *0*› **by** *auto*
show *?case* **using** ‹*a* > *0*› **and** $*$ **unfolding** *f'-def*
by (*auto simp*: *case-prod-unfold add-eq-0-iff intro*!: *continuous-intros*)
next
case (*4 b c z e*)
have $-$ *2* $*$ *real n* < σ **by** (*fact* σ)
also from *4 assms* **have** σ < *Re z* **by** *auto*
finally show *?case* **using** *assms 4*

by (*intro integrable-continuous-real continuous-intros*) (*auto simp*: *add-eq-0-iff*)
next
case (*5 t x s*)
thus *?case* **using** ‹*a* > *0*›
by (*intro integrable-EM-remainder′*) (*auto intro*!: *continuous-intros simp*: *add-eq-0-iff*)
next
case *6*
from *σ* **have** (*λy. C* / (−*2* ∗ *real n* − *σ*) ∗ (*a* + *y*) *powr* (−*2* ∗ *real n* − *σ*)) ⟶ *0*
by (*intro tendsto-mult-right-zero tendsto-neg-powr filterlim-real-sequentially filterlim-tendsto-add-at-top* [*OF tendsto-const*]) *auto*
thus *?case* **unfolding** *convergent-def* **by** (*auto simp*: *add-ac*)
next
case *7*
show *?case*
proof (*intro eventually-mono* [*OF eventually-ge-at-top*[*of 1*]] *ballI*)
fix *x* :: *real* **and** *s* :: *complex* **assume** *x*: *x* ≥ *1* **and** *s*: *s* ∈ *U*
have *norm* (− (*pochhammer s* (*Suc* (*2* ∗ *n*)) ∗ *of-real* (*x* + *a*) *powr* (− *1* − *2* ∗ *of-nat n* − *s*))) =
norm (*pochhammer s* (*Suc* (*2* ∗ *n*))) ∗ (*x* + *a*) *powr* (−*1* − *2* ∗ *of-nat n* − *Re s*)
(**is** *?N* = -) **using** *7* ‹*a* > *0*› *x* **by** (*simp add*: *norm-mult norm-powr-real-powr*)
also have . . . ≤ *?g x*
using *7 assms x s* ‹*a* > *0*› **by** (*intro mult-mono C powr-mono*) *auto*
finally show *?N* ≤ *?g x* **.**
qed
qed (*insert assms*, *auto*)
qed (*insert assms*, *auto*)

lemma *analytic-pre-zeta-aux*:
assumes *a* > *0*
shows *pre-zeta-aux n a analytic-on* {*s. Re s* > − *2* ∗ *real n*}
unfolding *analytic-on-def*
proof
fix *s* **assume** *s*: *s* ∈ {*s. Re s* > − *2* ∗ *real n*}
define *σ* **where** *σ* = (*Re s* − *2* ∗ *real n*) / *2*
with *s* **have** *σ*: *σ* > − *2* ∗ *real n*
by (*simp add*: *σ-def field-simps*)
from *s* **have** *s′*: *s* ∈ {*s. Re s* > *σ*}
by (*auto simp*: *σ-def field-simps*)

have *open* {*s. Re s* > *σ*}
by (*rule open-halfspace-Re-gt*)
with *s′* **obtain** *ε* **where** *ε* > *0 ball s ε* ⊆ {*s. Re s* > *σ*}
unfolding *open-contains-ball* **by** *blast*
with *σ* **have** *pre-zeta-aux n a holomorphic-on ball s ε*
by (*intro holomorphic-pre-zeta-aux′* [*OF assms*, *of - σ*]) *auto*

```
  with ‹ε > 0› show ∃e>0. pre-zeta-aux n a holomorphic-on ball s e
    by blast
qed
end

context
  fixes s :: complex and N :: nat and ζ :: complex ⇒ complex and a :: real
  assumes s: Re s > 1 and a: a > 0
  defines ζ ≡ (λs. ∑n. (of-nat n + of-real a) powr −s)
begin

interpretation ζ: euler-maclaurin-nat'
  λx. of-real (x + a) powr (1 − s) / (1 − s) λx. of-real (x + a) powr −s
  λn x. (−1) ^ n ∗ pochhammer s n ∗ of-real (x + a) powr −(s + n)
  0 N ζ s {}
proof (standard, goal-cases)
  case 2
  show ?case by (simp add: powr-minus field-simps)
next
  case (3 k)
  have complex-of-real x + complex-of-real a = 0 ⟷ x = −a for x
    by (simp only: of-real-add [symmetric] of-real-eq-0-iff add-eq-0-iff2)
  with a s show ?case
    by (intro continuous-intros) (auto simp: add-nonneg-nonneg)
next
  case (4 k x)
  with a have 0 < x + a by simp
  hence ∗: complex-of-real x + complex-of-real a ∉ ℝ≤0
    using nonpos-Reals-of-real-iff[of x+a, unfolded of-real-add] by auto
   have ∗∗: pochhammer z (Suc n) = − pochhammer z n ∗ (−z − of-nat n ::
complex) for z n
    by (simp add: pochhammer-rec' field-simps)
  show ((λx. (− 1) ^ k ∗ pochhammer s k ∗ of-real (x + a) powr − (s + of-nat
k))
          has-vector-derivative (− 1) ^ Suc k ∗ pochhammer s (Suc k) ∗
            of-real (x + a) powr − (s + of-nat (Suc k))) (at x)
     by (insert 4 ∗, (rule has-vector-derivative-real-field derivative-eq-intros refl |
simp)+)
       (auto simp: divide-simps powr-add powr-diff powr-minus ∗∗)
next
  case 5
  with s a show ?case
    by (auto intro!: continuous-intros simp: minus-equation-iff add-eq-0-iff)
next
  case (6 x)
  with a have 0 < x + a by simp
  hence ∗: complex-of-real x + complex-of-real a ∉ ℝ≤0
    using nonpos-Reals-of-real-iff[of x+a, unfolded of-real-add] by auto
  show ?case unfolding of-real-add
```

by (*insert 6 s* ∗, (*rule has-vector-derivative-real-field derivative-eq-intros refl* | *force simp add*: *minus-equation-iff*)+)
next
case *7*
from *s a* **have** (λk. (*of-nat k* + *of-real a*) *powr* −*s*) *sums* ζ *s*
unfolding ζ*-def* **by** (*intro summable-sums summable-hurwitz-zeta*) *auto*
hence *1*: (λb. ($\sum$ *k*=*0*..*b*. (*of-nat k* + *of-real a*) *powr* −*s*)) ⟶ ζ *s*
by (*simp add*: *sums-def'*)

{
fix *z* **assume** *Re z* < *0*
hence ((λb. (*a* + *real b*) *powr Re z*) ⟶ *0*) *at-top*
by (*intro tendsto-neg-powr filterlim-tendsto-add-at-top filterlim-real-sequentially*) *auto*
also have (λb. (*a* + *real b*) *powr Re z*) = (λb. *norm* ((*of-nat b* + *a*) *powr z*))
using *a* **by** (*subst norm-powr-real-powr*) (*auto simp*: *add-ac*)
finally have ((λb. (*of-nat b* + *a*) *powr z*) ⟶ *0*) *at-top*
by (*subst* (*asm*) *tendsto-norm-zero-iff*) *simp*
} **note** ∗ = *this*
have (λb. (*of-nat b* + *a*) *powr* (*1* − *s*) / (*1* − *s*)) ⟶ *0* / (*1* − *s*)
using *s* **by** (*intro tendsto-divide tendsto-const* ∗) *auto*
hence *2*: (λb. (*of-nat b* + *a*) *powr* (*1* − *s*) / (*1* − *s*)) ⟶ *0*
by *simp*

have (λb. ($\sum$ *i*<*2* ∗ *N* + *1*. (*bernoulli'* (*Suc i*) / *fact* (*Suc i*)) $*_R$ ((− *1*) ^ *i* ∗ *pochhammer s i* ∗ (*of-nat b* + *a*) *powr* −(*s* + *of-nat i*)))) ⟶ ($\sum$ *i*<*2* ∗ *N* + *1*. (*bernoulli'* (*Suc i*) / *fact* (*Suc i*)) $*_R$ ((− *1*) ^ *i* ∗ *pochhammer s i* ∗ *0*))
using *s* **by** (*intro tendsto-intros* ∗) *auto*
hence *3*: (λb. ($\sum$ *i*<*2* ∗ *N* + *1*. (*bernoulli'* (*Suc i*) / *fact* (*Suc i*)) $*_R$ ((− *1*) ^ *i* ∗ *pochhammer s i* ∗ (*of-nat b* + *a*) *powr* −(*s* + *of-nat i*)))) ⟶ *0*
by *simp*

from *tendsto-diff*[*OF tendsto-diff*[*OF 1 2*] *3*]
show *?case* **by** *simp*
qed *simp-all*

The pre-ζ functions agree with the infinite sum that is used to define the ζ function for $\Re(s) > 1$.

lemma *pre-zeta-aux-conv-zeta*:
pre-zeta-aux N a s = ζ *s* + *a powr* (*1* − *s*) / (*1* − *s*)
proof −
let *?R* = ($\sum$ *i*=*1*..*N*. ((*bernoulli* (*2*∗*i*) / *fact* (*2*∗*i*)) $*_R$ *pochhammer s* (*2*∗*i* − *1*) ∗ *of-real a powr* (−*s* − (*2*∗*i*−*1*))))
let *?S* = *EM-remainder* (*Suc* (*2* ∗ *N*)) (λx. − (*pochhammer s* (*Suc* (*2*∗*N*)) ∗ *of-real* (*x* + *a*) *powr* (− *1* − *2* ∗ *of-nat N* − *s*))) *0*
from ζ.*euler-maclaurin-strong-nat'*[*OF le-refl*, *simplified*]
have *of-real a powr* −*s* = *a powr* (*1* − *s*) / (*1* − *s*) + ζ *s* + *a powr* −*s* / *2* +

$(-?R) - ?S$
  unfolding *sum-negf* [*symmetric*] **by** (*simp add*: *scaleR-conv-of-real pre-zeta-aux-def mult-ac*)
 thus *?thesis* **unfolding** *pre-zeta-aux-def*

  by (*simp add*: *field-simps del*: *div-mult-self3 div-mult-self4 div-mult-self2 div-mult-self1*)
qed

end

Since all of the partial pre-ζ functions are analytic and agree in the halfspace with $\mathfrak{R}(s) > 0$, they must agree in their entire domain.

lemma *pre-zeta-aux-eq*:
 assumes $m \leq n$ $a > 0$ *Re* $s > -2 * $ *real* m
 shows *pre-zeta-aux m a s* = *pre-zeta-aux n a s*
proof −
 have *pre-zeta-aux n a s* − *pre-zeta-aux m a s* = *0*
 proof (*rule analytic-continuation*[*of* λ*s*. *pre-zeta-aux n a s* − *pre-zeta-aux m a s*])
  show (λ*s*. *pre-zeta-aux n a s* − *pre-zeta-aux m a s*) *holomorphic-on* {*s*. *Re s* > −*2* ∗ *real m*}
   using *assms* **by** (*intro holomorphic-intros analytic-imp-holomorphic analytic-on-subset*[*OF analytic-pre-zeta-aux*]) *auto*
 next
  fix *s* **assume** *s* ∈ {*s*. *Re s* > *1*}
  with ‹*a* > *0*› **show** *pre-zeta-aux n a s* − *pre-zeta-aux m a s* = *0*
   by (*simp add*: *pre-zeta-aux-conv-zeta*)
 next
  have *2* ∈ {*s*. *Re s* > *1*} **by** *simp*
  also have ... = *interior* ...
   by (*intro interior-open* [*symmetric*] *open-halfspace-Re-gt*)
  finally show *2 islimpt* {*s*. *Re s* > *1*}
   by (*rule interior-limit-point*)
 next
  show *connected* {*s*. *Re s* > −*2* ∗ *real m*}
   using *convex-halfspace-gt*[*of* −*2* ∗ *real m 1*::*complex*]
   by (*intro convex-connected*) *auto*
 qed (*insert assms*, *auto simp*: *open-halfspace-Re-gt*)
 thus *?thesis* **by** *simp*
qed

lemma *pre-zeta-aux-eq'*:
 assumes *a* > *0* *Re s* > −*2* ∗ *real m* *Re s* > −*2* ∗ *real n*
 shows *pre-zeta-aux m a s* = *pre-zeta-aux n a s*
proof (*cases m n rule*: *linorder-cases*)
 case *less*
 with *assms* **show** *?thesis* **by** (*intro pre-zeta-aux-eq*) *auto*
next
 case *greater*

with *assms* **show** *?thesis* **by** (*subst eq-commute*, *intro pre-zeta-aux-eq*) *auto*
qed *auto*

lemma *pre-zeta-aux-eq-pre-zeta*:
assumes $Re\ s > -2 * real\ n$ **and** $a > 0$
shows *pre-zeta-aux n a s = pre-zeta a s*
unfolding *pre-zeta-def*
proof (*intro pre-zeta-aux-eq'*)
from *assms* **show** $- 2 * real\ (nat\ (1 - \lceil Re\ s\ /\ 2\rceil)) < Re\ s$
by *linarith*
qed (*insert assms*, *simp-all*)

This means that the idea of iterating that construction infinitely does yield a well-defined entire function.

lemma *analytic-pre-zeta*:
assumes $a > 0$
shows *pre-zeta a analytic-on A*
unfolding *analytic-on-def*
proof
fix *s* **assume** $s \in A$
let $?B = \{s'.\ Re\ s' > of\text{-}int\ \lfloor Re\ s\rfloor - 1\}$
have *s*: $s \in ?B$ **by** *simp linarith?*
moreover have *open ?B* **by** (*rule open-halfspace-Re-gt*)
ultimately obtain ε **where** ε: $\varepsilon > 0$ *ball s* $\varepsilon \subseteq ?B$
unfolding *open-contains-ball* **by** *blast*
define *C* **where** $C = ball\ s\ \varepsilon$

note *analytic = analytic-on-subset*[*OF analytic-pre-zeta-aux*]
have *pre-zeta-aux* $(nat\ \lceil- Re\ s\rceil + 2)$ *a holomorphic-on C*
proof (*intro analytic-imp-holomorphic analytic subsetI assms*, *goal-cases*)
case (*1 w*)
with ε **have** $w \in ?B$ **by** (*auto simp*: *C-def*)
thus *?case* **by** (*auto simp*: *ceiling-minus*)
qed
also have *?this* $\longleftrightarrow$ *pre-zeta a holomorphic-on C*
proof (*intro holomorphic-cong refl pre-zeta-aux-eq-pre-zeta assms*)
fix *w* **assume** $w \in C$
with ε **have** *w*: $w \in ?B$ **by** (*auto simp*: *C-def*)
thus $- 2 * real\ (nat\ \lceil- Re\ s\rceil + 2) < Re\ w$
by (*simp add*: *ceiling-minus*)
qed
finally show $\exists\ e>0.$ *pre-zeta a holomorphic-on ball s e*
using ‹$\varepsilon > 0$› **unfolding** *C-def* **by** *blast*
qed

lemma *holomorphic-pre-zeta* [*holomorphic-intros*]:
f holomorphic-on A $\Longrightarrow a > 0 \Longrightarrow (\lambda z.\ pre\text{-}zeta\ a\ (f\ z))$ *holomorphic-on A*
using *holomorphic-on-compose* [*OF - analytic-imp-holomorphic* [*OF analytic-pre-zeta*], *of f*]

by (*simp add*: *o-def*)

corollary *continuous-on-pre-zeta*:
  $a > 0 \Longrightarrow$ *continuous-on A* (*pre-zeta a*)
  by (*intro holomorphic-on-imp-continuous-on holomorphic-intros*) *auto*

corollary *continuous-on-pre-zeta′* [*continuous-intros*]:
  continuous-on A f $\Longrightarrow a > 0 \Longrightarrow$ *continuous-on A* (λx. *pre-zeta a* (*f x*))
  using *continuous-on-compose2* [*OF continuous-on-pre-zeta*, *of a A f f ‘ A*]
  by (*auto simp*: *image-iff*)

corollary *continuous-pre-zeta* [*continuous-intros*]:
  $a > 0 \Longrightarrow$ *continuous* (*at s within A*) (*pre-zeta a*)
  by (*rule continuous-within-subset*[*of - UNIV*])
    (*insert continuous-on-pre-zeta*[*of a UNIV*],
     auto simp: *continuous-on-eq-continuous-at open-Compl*)

corollary *continuous-pre-zeta′* [*continuous-intros*]:
  $a > 0 \Longrightarrow$ *continuous* (*at s within A*) *f* $\Longrightarrow$
    continuous (*at s within A*) (λs. *pre-zeta a* (*f s*))
  using *continuous-within-compose3*[*OF continuous-pre-zeta*, *of a s A f*] **by** *auto*

It is now obvious that ζ is holomorphic everywhere except 1, where it has a simple pole with residue 1, which we can simply read off.

theorem *holomorphic-hurwitz-zeta*:
  assumes $a > 0$ $1 \notin A$
  shows *hurwitz-zeta a holomorphic-on A*
proof −
  have (λs. *pre-zeta a s* + *complex-of-real a powr* ($1 - s$) / ($s - 1$)) *holomorphic-on A*
    using *assms* **by** (*auto intro*!: *holomorphic-intros*)
  also from *assms* **have** *?this* $\longleftrightarrow$ *?thesis*
    by (*intro holomorphic-cong*) (*auto simp*: *hurwitz-zeta-def*)
  finally show *?thesis* **.**
qed

corollary *holomorphic-hurwitz-zeta′* [*holomorphic-intros*]:
  assumes *f holomorphic-on A* **and** $a > 0$ **and** $\bigwedge z.\ z \in A \Longrightarrow f\ z \neq 1$
  shows (λx. *hurwitz-zeta a* (*f x*)) *holomorphic-on A*
proof −
  have *hurwitz-zeta a* $\circ$ *f holomorphic-on A* **using** *assms*
    by (*intro holomorphic-on-compose-gen*[*of - - - f ‘ A*] *holomorphic-hurwitz-zeta assms*) *auto*
  thus *?thesis* **by** (*simp add*: *o-def*)
qed

theorem *holomorphic-zeta*: $1 \notin A \Longrightarrow$ *zeta holomorphic-on A*
  unfolding *zeta-def* **by** (*auto intro*: *holomorphic-intros*)

```
corollary holomorphic-zeta' [holomorphic-intros]:
  assumes f holomorphic-on A and ⋀z. z ∈ A ⟹ f z ≠ 1
  shows   (λx. zeta (f x)) holomorphic-on A
  using assms unfolding zeta-def by (auto intro: holomorphic-intros)

corollary analytic-hurwitz-zeta:
  assumes a > 0 1 ∉ A
  shows   hurwitz-zeta a analytic-on A
proof −
  from assms(1) have hurwitz-zeta a holomorphic-on −{1}
    by (rule holomorphic-hurwitz-zeta) auto
  also have ?this ⟷ hurwitz-zeta a analytic-on −{1}
    by (intro analytic-on-open [symmetric]) auto
  finally show ?thesis by (rule analytic-on-subset) (insert assms, auto)
qed

corollary analytic-zeta: 1 ∉ A ⟹ zeta analytic-on A
  unfolding zeta-def by (rule analytic-hurwitz-zeta) auto

corollary continuous-on-hurwitz-zeta:
  a > 0 ⟹ 1 ∉ A ⟹ continuous-on A (hurwitz-zeta a)
  by (intro holomorphic-on-imp-continuous-on holomorphic-intros) auto

corollary continuous-on-hurwitz-zeta' [continuous-intros]:
  continuous-on A f ⟹ a > 0 ⟹ (⋀x. x ∈ A ⟹ f x ≠ 1) ⟹
     continuous-on A (λx. hurwitz-zeta a (f x))
  using continuous-on-compose2 [OF continuous-on-hurwitz-zeta, of a f ‘ A A f]
  by (auto simp: image-iff)

corollary continuous-on-zeta: 1 ∉ A ⟹ continuous-on A zeta
  by (intro holomorphic-on-imp-continuous-on holomorphic-intros) auto

corollary continuous-on-zeta' [continuous-intros]:
  continuous-on A f ⟹ (⋀x. x ∈ A ⟹ f x ≠ 1) ⟹
     continuous-on A (λx. zeta (f x))
  using continuous-on-compose2 [OF continuous-on-zeta, of f ‘ A A f]
  by (auto simp: image-iff)

corollary continuous-hurwitz-zeta [continuous-intros]:
  a > 0 ⟹ s ≠ 1 ⟹ continuous (at s within A) (hurwitz-zeta a)
  by (rule continuous-within-subset[of - UNIV])
     (insert continuous-on-hurwitz-zeta[of a −{1}],
      auto simp: continuous-on-eq-continuous-at open-Compl)

corollary continuous-hurwitz-zeta' [continuous-intros]:
  a > 0 ⟹ f s ≠ 1 ⟹ continuous (at s within A) f ⟹
     continuous (at s within A) (λs. hurwitz-zeta a (f s))
  using continuous-within-compose3[OF continuous-hurwitz-zeta, of a f s A] by
auto
```

corollary *continuous-zeta* [*continuous-intros*]:
s ≠ *1* ⟹ *continuous* (*at s within A*) *zeta*
unfolding *zeta-def* **by** (*intro continuous-intros*) *auto*

corollary *continuous-zeta′* [*continuous-intros*]:
f s ≠ *1* ⟹ *continuous* (*at s within A*) *f* ⟹ *continuous* (*at s within A*) (λ*s*. *zeta* (*f s*))
unfolding *zeta-def* **by** (*intro continuous-intros*) *auto*

corollary *field-differentiable-at-zeta*:
assumes *s* ≠ *1*
shows *zeta field-differentiable at s*
proof −
have *zeta holomorphic-on* (− {*1*}) **using** *holomorphic-zeta* **by** *force*
moreover have *open* (−{*1*} :: *complex set*) **by** (*intro open-Compl*) *auto*
ultimately show *?thesis* **using** *assms*
by (*auto simp add*: *holomorphic-on-open open-halfspace-Re-gt open-Diff field-differentiable-def*)
qed

theorem *is-pole-hurwitz-zeta*:
assumes *a* > *0*
shows *is-pole* (*hurwitz-zeta a*) *1*
proof −
from *assms* **have** *continuous-on UNIV* (*pre-zeta a*)
by (*intro holomorphic-on-imp-continuous-on analytic-imp-holomorphic analytic-pre-zeta*)
hence *isCont* (*pre-zeta a*) *1*
by (*auto simp*: *continuous-on-eq-continuous-at*)
hence ∗: *pre-zeta a* −*1*→ *pre-zeta a 1*
by (*simp add*: *isCont-def*)
from *assms* **have** *isCont* (λ*s*. *complex-of-real a powr* (*1* − *s*)) *1*
by (*intro isCont-powr-complex*) *auto*
with *assms* **have** ∗∗: (λ*s*. *complex-of-real a powr* (*1* − *s*)) −*1*→ *1*
by (*simp add*: *isCont-def*)
have (λ*s*::*complex*. *s* − *1*) −*1*→ *1* − *1* **by** (*intro tendsto-intros*)
hence *filterlim* (λ*s*::*complex*. *s* − *1*) (*at 0*) (*at 1*)
by (*auto simp*: *filterlim-at eventually-at-filter*)
hence ∗∗∗: *filterlim* (λ*s* :: *complex*. *a powr* (*1* − *s*) / (*s* − *1*)) *at-infinity* (*at 1*)
by (*intro filterlim-divide-at-infinity* [*OF* ∗∗]) *auto*
have *is-pole* (λ*s*. *pre-zeta a s* + *complex-of-real a powr* (*1* − *s*) / (*s* − *1*)) *1*
unfolding *is-pole-def hurwitz-zeta-def* **by** (*rule tendsto-add-filterlim-at-infinity* ∗ ∗∗∗)+
also have *?this* ⟷ *?thesis* **unfolding** *is-pole-def*
by (*intro filterlim-cong refl*) (*auto simp*: *eventually-at-filter hurwitz-zeta-def*)
finally show *?thesis* **.**
qed

corollary *is-pole-zeta*: *is-pole zeta 1*

unfolding *zeta-def* **by** (*rule is-pole-hurwitz-zeta*) *auto*

theorem *zorder-hurwitz-zeta*:
assumes *a* > *0*
shows *zorder* (*hurwitz-zeta a*) *1* = −*1*
proof (*rule zorder-eqI*[*of UNIV*])
fix *w* :: *complex* **assume** *w* ∈ *UNIV w* ≠ *1*
thus *hurwitz-zeta a w* = (*pre-zeta a w* ∗ (*w* − *1*) + *a powr* (*1* − *w*)) ∗ (*w* − *1*) *powr* (*of-int* (−*1*))
apply (*subst* (*1*) *powr-of-int*)
by (*auto simp add*: *hurwitz-zeta-def field-simps*)
qed (*insert assms*, *auto intro*!: *holomorphic-intros*)

corollary *zorder-zeta*: *zorder zeta 1* = − *1*
unfolding *zeta-def* **by** (*rule zorder-hurwitz-zeta*) *auto*

theorem *residue-hurwitz-zeta*:
assumes *a* > *0*
shows *residue* (*hurwitz-zeta a*) *1* = *1*
proof −
note *holo* = *analytic-imp-holomorphic*[*OF analytic-pre-zeta*]
have *residue* (*hurwitz-zeta a*) *1* = *residue* (λ*z*. *pre-zeta a z* + *a powr* (*1* − *z*) / (*z* − *1*)) *1*
by (*intro residue-cong*) (*auto simp*: *eventually-at-filter hurwitz-zeta-def*)
also have . . . = *residue* (λ*z*. *a powr* (*1* − *z*) / (*z* − *1*)) *1* **using** *assms*
by (*subst residue-add* [*of UNIV*])
(*auto intro*!: *holomorphic-intros holo intro*: *residue-holo*[*of UNIV*, *OF* - - *holo*])
also have . . . = *complex-of-real a powr* (*1* − *1*)
using *assms* **by** (*intro residue-simple* [*of UNIV*]) (*auto intro*!: *holomorphic-intros*)
also from *assms* **have** . . . = *1* **by** *simp*
finally show *?thesis* **.**
qed

corollary *residue-zeta*: *residue zeta 1* = *1*
unfolding *zeta-def* **by** (*rule residue-hurwitz-zeta*) *auto*

lemma *zeta-bigo-at-1*: *zeta* ∈ *O*[*at 1 within A*](λ*x*. *1* / (*x* − *1*))
proof −
have *zeta* ∈ Θ[*at 1 within A*](λ*s*. *pre-zeta 1 s* + *1* / (*s* − *1*))
by (*intro bigthetaI-cong*) (*auto simp*: *eventually-at-filter zeta-def hurwitz-zeta-def*)
also have (λ*s*. *pre-zeta 1 s* + *1* / (*s* − *1*)) ∈ *O*[*at 1 within A*](λ*s*. *1* / (*s* − *1*))
proof (*rule sum-in-bigo*)
have *continuous-on UNIV* (*pre-zeta 1*)
by (*intro holomorphic-on-imp-continuous-on holomorphic-intros*) *auto*
hence *isCont* (*pre-zeta 1*) *1* **by** (*auto simp*: *continuous-on-eq-continuous-at*)
hence *continuous* (*at 1 within A*) (*pre-zeta 1*)
by (*rule continuous-within-subset*) *auto*
hence *pre-zeta 1* ∈ *O*[*at 1 within A*](λ-. *1*)

by (*intro continuous-imp-bigo-1*) *auto*
also have *ev*: *eventually* ($\lambda s.$ $s \in$ *ball 1 1* $\wedge$ $s \neq 1$ $\wedge$ $s \in A$) (*at 1 within A*)
by (*intro eventually-at-ball′*) *auto*
have (λ-. *1*) $\in$ *O*[*at 1 within A*]($\lambda s.$ *1* / (s − *1*))
by (*intro landau-o.bigI*[*of 1*] *eventually-mono*[*OF ev*])
(*auto simp*: *eventually-at-filter norm-divide dist-norm norm-minus-commute field-simps*)
finally show *pre-zeta 1* $\in$ *O*[*at 1 within A*]($\lambda s.$ *1* / (s − *1*)) **.**
qed *simp-all*
finally show *?thesis* **.**
qed

theorem
assumes $a > 0$ *Re* $s > 1$
shows *hurwitz-zeta-conv-suminf*: *hurwitz-zeta a s* = ($\sum n.$ (*of-nat n* + *of-real a*) *powr* −s)
and *sums-hurwitz-zeta*: ($\lambda n.$ (*of-nat n* + *of-real a*) *powr* −s) *sums hurwitz-zeta a s*
proof −
from *assms* **have** [*simp*]: $s \neq 1$ **by** *auto*
from *assms* **have** *hurwitz-zeta a s* = *pre-zeta-aux 0 a s* + *of-real a powr* (*1* − s) / (s − *1*)
by (*simp add*: *hurwitz-zeta-def pre-zeta-def*)
also from *assms* **have** *pre-zeta-aux 0 a s* = ($\sum n.$ (*of-nat n* + *of-real a*) *powr* −s) +
of-real a powr (*1* − s) / (*1* − s)
by (*intro pre-zeta-aux-conv-zeta*)
also have ... + *a powr* (*1* − s) / (s − *1*) =
($\sum n.$ (*of-nat n* + *of-real a*) *powr* −s) + *a powr* (*1* − s) ∗ (*1* / (*1* − s) + *1* / (s − *1*))
by (*simp add*: *algebra-simps*)
also have *1* / (*1* − s) + *1* / (s − *1*) = *0*
by (*simp add*: *divide-simps*)
finally show *hurwitz-zeta a s* = ($\sum n.$ (*of-nat n* + *of-real a*) *powr* −s) **by** *simp*
moreover have ($\lambda n.$ (*of-nat n* + *of-real a*) *powr* −s) *sums* ($\sum n.$ (*of-nat n* + *of-real a*) *powr* −s)
by (*intro summable-sums summable-hurwitz-zeta assms*)
ultimately show ($\lambda n.$ (*of-nat n* + *of-real a*) *powr* −s) *sums hurwitz-zeta a s*
by *simp*
qed

corollary
assumes *Re* $s > 1$
shows *zeta-conv-suminf*: *zeta s* = ($\sum n.$ *of-nat* (*Suc n*) *powr* −s)
and *sums-zeta*: ($\lambda n.$ *of-nat* (*Suc n*) *powr* −s) *sums zeta s*
using *hurwitz-zeta-conv-suminf*[*of 1 s*] *sums-hurwitz-zeta*[*of 1 s*] *assms*
by (*simp-all add*: *zeta-def add-ac*)

corollary

assumes *n* > *1*
shows *zeta-nat-conv-suminf*: *zeta* (*of-nat n*) = ($\sum$*k*. *1* / *of-nat* (*Suc k*) ^ *n*)
and *sums-zeta-nat*: (λ*k*. *1* / *of-nat* (*Suc k*) ^ *n*) *sums zeta* (*of-nat n*)
proof −
have (λ*k*. *of-nat* (*Suc k*) *powr* −*of-nat n*) *sums zeta* (*of-nat n*)
using *assms* **by** (*intro sums-zeta*) *auto*
also have (λ*k*. *of-nat* (*Suc k*) *powr* −*of-nat n*) = (λ*k*. *1* / *of-nat* (*Suc k*) ^ *n* :: *complex*)
by (*simp add*: *powr-minus divide-simps del*: *of-nat-Suc*)
finally show (λ*k*. *1* / *of-nat* (*Suc k*) ^ *n*) *sums zeta* (*of-nat n*) **.**
thus *zeta* (*of-nat n*) = ($\sum$*k*. *1* / *of-nat* (*Suc k*) ^ *n*) **by** (*simp add*: *sums-iff*)
qed

lemma *pre-zeta-aux-cnj* [*simp*]:
assumes *a* > *0*
shows *pre-zeta-aux n a* (*cnj z*) = *cnj* (*pre-zeta-aux n a z*)
proof −
have *cnj* (*pre-zeta-aux n a z*) =
of-real a powr −*cnj z* / *2* + ($\sum$*x*=*1*..*n*. (*bernoulli* (*2* ∗ *x*) / *fact* (*2* ∗ *x*)) $*_R$
a powr (−*cnj z* − (*2*∗*x*−*1*)) ∗ *pochhammer* (*cnj z*) (*2*∗*x*−*1*)) + *EM-remainder* (*2*∗*n*+*1*)
(λ*x*. −(*pochhammer* (*cnj z*) (*Suc* (*2* ∗ *n*)) ∗
cnj (*of-real* (*x* + *a*) *powr* (−*1* − *2* ∗ *of-nat n* − *z*)))) *0*
(**is** - = - + *?A* + *?B*) **unfolding** *pre-zeta-aux-def complex-cnj-add* **using** *assms*
by (*subst EM-remainder-cnj* [*symmetric*])
(*auto intro*!: *continuous-intros simp*: *cnj-powr add-eq-0-iff mult-ac*)
also have *?B* = *EM-remainder* (*2*∗*n*+*1*)
(λ*x*. −(*pochhammer* (*cnj z*) (*Suc* (*2* ∗ *n*)) ∗ *of-real* (*x* + *a*) *powr* (−*1* − *2* ∗ *of-nat n* − *cnj z*))) *0*
using *assms* **by** (*intro EM-remainder-cong*) (*auto simp*: *cnj-powr*)
also have *of-real a powr* −*cnj z* / *2* + *?A* + ... = *pre-zeta-aux n a* (*cnj z*)
by (*simp add*: *pre-zeta-aux-def mult-ac*)
finally show *?thesis* **..**
qed

lemma *pre-zeta-cnj* [*simp*]: *a* > *0* ⟹ *pre-zeta a* (*cnj z*) = *cnj* (*pre-zeta a z*)
by (*simp add*: *pre-zeta-def*)

lemma *hurwitz-zeta-cnj* [*simp*]: *a* > *0* ⟹ *hurwitz-zeta a* (*cnj z*) = *cnj* (*hurwitz-zeta a z*)
proof −
assume *a* > *0*
moreover have *cnj z* = *1* ⟷ *z* = *1* **by** (*simp add*: *complex-eq-iff*)
ultimately show *?thesis* **by** (*auto simp*: *hurwitz-zeta-def cnj-powr*)
qed

lemma *zeta-cnj* [*simp*]: *zeta* (*cnj z*) = *cnj* (*zeta z*)
by (*simp add*: *zeta-def*)

corollary *hurwitz-zeta-real*: $a > 0 \Longrightarrow$ *hurwitz-zeta a* (*of-real x*) $\in \mathbb{R}$
using *hurwitz-zeta-cnj* [*of a of-real x*] **by** (*simp add*: *Reals-cnj-iff del*: *zeta-cnj*)

corollary *zeta-real*: *zeta* (*of-real x*) $\in \mathbb{R}$
unfolding *zeta-def* **by** (*rule hurwitz-zeta-real*) *auto*

corollary *zeta-real'*: $z \in \mathbb{R} \Longrightarrow$ *zeta z* $\in \mathbb{R}$
by (*elim Reals-cases*) (*auto simp*: *zeta-real*)

2.3 Connection to Dirichlet series

lemma *eval-fds-zeta*: *Re s* $> 1 \Longrightarrow$ *eval-fds fds-zeta s* = *zeta s*
using *sums-zeta* [*of s*] **by** (*intro eval-fds-eqI*) (*auto simp*: *powr-minus divide-simps*)

theorem *euler-product-zeta*:
assumes *Re s* > 1
shows ($\lambda n.$ $\prod p \le n.$ *if prime p then inverse* ($1 - 1$ / *of-nat p powr s*) *else* 1) $\longrightarrow$ *zeta s*
using *euler-product-fds-zeta*[*of s*] *assms* **unfolding** *nat-power-complex-def*
by (*simp add*: *eval-fds-zeta*)

corollary *euler-product-zeta'*:
assumes *Re s* > 1
shows ($\lambda n.$ $\prod p \mid$ *prime p* $\wedge p \le n.$ *inverse* ($1 - 1$ / *of-nat p powr s*)) $\longrightarrow$ *zeta s*
proof −
note *euler-product-zeta* [*OF assms*]
also have ($\lambda n.$ $\prod p \le n.$ *if prime p then inverse* ($1 - 1$ / *of-nat p powr s*) *else* 1) =
($\lambda n.$ $\prod p \mid$ *prime p* $\wedge p \le n.$ *inverse* ($1 - 1$ / *of-nat p powr s*))
by (*intro ext prod.mono-neutral-cong-right refl*) *auto*
finally show *?thesis* **.**
qed

theorem *zeta-Re-gt-1-nonzero*: *Re s* $> 1 \Longrightarrow$ *zeta s* $\neq 0$
using *eval-fds-zeta-nonzero*[*of s*] **by** (*simp add*: *eval-fds-zeta*)

theorem *tendsto-zeta-Re-going-to-at-top*: (*zeta* $\longrightarrow 1$) (*Re going-to at-top*)
proof (*rule Lim-transform-eventually*)
have *eventually* (λx::*real*. $x > 1$) *at-top*
by (*rule eventually-gt-at-top*)
hence *eventually* ($\lambda s.$ *Re s* > 1) (*Re going-to at-top*)
by *blast*
thus *eventually* ($\lambda z.$ *eval-fds fds-zeta z* = *zeta z*) (*Re going-to at-top*)
by *eventually-elim* (*simp add*: *eval-fds-zeta*)
next
have *conv-abscissa* (*fds-zeta* :: *complex fds*) ≤ 1
proof (*rule conv-abscissa-leI*)

fix c' **assume** *ereal* $c' > 1$
thus $\exists s.\ s \cdot 1 = c' \wedge$ *fds-converges fds-zeta* (*s::complex*)
by (*auto intro*!: *exI*[*of - of-real c'*])
qed
hence (*eval-fds fds-zeta* $\longrightarrow$ *fds-nth fds-zeta 1*) (*Re going-to at-top*)
by (*intro tendsto-eval-fds-Re-going-to-at-top'*) *auto*
thus (*eval-fds fds-zeta* $\longrightarrow$ *1*) (*Re going-to at-top*) **by** *simp*
qed

lemma *conv-abscissa-zeta* [*simp*]: *conv-abscissa* (*fds-zeta* :: *complex fds*) = *1*
and *abs-conv-abscissa-zeta* [*simp*]: *abs-conv-abscissa* (*fds-zeta* :: *complex fds*) = *1*
proof −
let *?z* = *fds-zeta* :: *complex fds*
have *A*: *conv-abscissa ?z* $\geq$ *1*
proof (*intro conv-abscissa-geI*)
fix c' **assume** *ereal* $c' < 1$
hence $\neg$*summable* ($\lambda n.$ *real n powr* $-c'$)
by (*subst summable-real-powr-iff*) *auto*
hence $\neg$*summable* ($\lambda n.$ *of-real* (*real n powr* $-c'$) :: *complex*)
by (*subst summable-of-real-iff*)
also have *summable* ($\lambda n.$ *of-real* (*real n powr* $-c'$) :: *complex*) $\longleftrightarrow$
fds-converges fds-zeta (*of-real* c' :: *complex*)
unfolding *fds-converges-def*
by (*intro summable-cong eventually-mono* [*OF eventually-gt-at-top*[*of 0*]])
(*simp add*: *fds-nth-zeta powr-Reals-eq powr-minus divide-simps*)
finally show $\exists$ *s::complex*. $s \cdot 1 = c' \wedge \neg$*fds-converges fds-zeta s*
by (*intro exI*[*of - of-real c'*]) *auto*
qed

have *B*: *abs-conv-abscissa ?z* $\leq$ *1*
proof (*intro abs-conv-abscissa-leI*)
fix c' **assume** $1 <$ *ereal* c'
thus $\exists$ *s::complex*. $s \cdot 1 = c' \wedge$ *fds-abs-converges fds-zeta s*
by (*intro exI*[*of - of-real c'*]) *auto*
qed

have *conv-abscissa ?z* $\leq$ *abs-conv-abscissa ?z*
by (*rule conv-le-abs-conv-abscissa*)
also note *B*
finally show *conv-abscissa ?z* = *1* **using** *A* **by** (*intro antisym*)

note *A*
also have *conv-abscissa ?z* $\leq$ *abs-conv-abscissa ?z*
by (*rule conv-le-abs-conv-abscissa*)
finally show *abs-conv-abscissa ?z* = *1* **using** *B* **by** (*intro antisym*)
qed

theorem *deriv-zeta-sums*:

assumes *s*: *Re s* > *1*
shows ($\lambda n.$ $-$*of-real* (*ln* (*real* (*Suc n*))) / *of-nat* (*Suc n*) *powr s*) *sums deriv zeta s*
proof $-$
from *s* **have** *fds-converges* (*fds-deriv fds-zeta*) *s*
by (*intro fds-converges-deriv*) *simp-all*
with *s* **have** ($\lambda n.$ $-$*of-real* (*ln* (*real* (*Suc n*))) / *of-nat* (*Suc n*) *powr s*) *sums deriv* (*eval-fds fds-zeta*) *s*
unfolding *fds-converges-altdef*
by (*simp add*: *fds-nth-deriv scaleR-conv-of-real eval-fds-deriv eval-fds-zeta*)
also from *s* **have** *eventually* ($\lambda s.$ $s \in \{s.$ *Re s* > *1*$\}$) (*nhds s*)
by (*intro eventually-nhds-in-open*) (*auto simp*: *open-halfspace-Re-gt*)
hence *eventually* ($\lambda s.$ *eval-fds fds-zeta s* = *zeta s*) (*nhds s*)
by *eventually-elim* (*auto simp*: *eval-fds-zeta*)
hence *deriv* (*eval-fds fds-zeta*) *s* = *deriv zeta s*
by (*intro deriv-cong-ev refl*)
finally show *?thesis* **.**
qed

theorem *inverse-zeta-sums*:
assumes *s*: *Re s* > *1*
shows ($\lambda n.$ *moebius-mu* (*Suc n*) / *of-nat* (*Suc n*) *powr s*) *sums inverse* (*zeta s*)
proof $-$
have *fds-converges* (*fds moebius-mu*) *s*
using *assms* **by** (*auto intro*!: *fds-abs-converges-moebius-mu*)
hence ($\lambda n.$ *moebius-mu* (*Suc n*) / *of-nat* (*Suc n*) *powr s*) *sums eval-fds* (*fds moebius-mu*) *s*
by (*simp add*: *fds-converges-altdef*)
also have *fds moebius-mu* = *inverse* (*fds-zeta* :: *complex fds*)
by (*rule fds-moebius-inverse-zeta*)
also from *s* **have** *eval-fds* ... *s* = *inverse* (*zeta s*)
by (*subst eval-fds-inverse*)
(*auto simp*: *fds-moebius-inverse-zeta* [*symmetric*] *eval-fds-zeta*
intro!: *fds-abs-converges-moebius-mu*)
finally show *?thesis* **.**
qed

The following gives an extension of the ζ functions to the critical strip.

lemma *hurwitz-zeta-critical-strip*:
fixes *s* :: *complex* **and** *a* :: *real*
defines $S \equiv (\lambda n.\ \sum i{<}n.\ (\textit{of-nat}\ i + a)\ \textit{powr} - s)$
defines $I' \equiv (\lambda n.\ \textit{of-nat}\ n\ \textit{powr}\ (1 - s) / (1 - s))$
assumes *Re s* > *0* $s \neq 1$ **and** $a > 0$
shows ($\lambda n.$ $S\ n - I'\ n$) $\longrightarrow$ *hurwitz-zeta a s*
proof $-$
from *assms* **have** [*simp*]: $s \neq 1$ **by** *auto*
let *?f* = $\lambda x.$ *of-real* ($x + a$) *powr* $-s$
let *?fs* = $\lambda n\ x.$ (-1) ^ n $*$ *pochhammer s n* $*$ *of-real* ($x + a$) *powr* ($-s$ $-$ *of-nat n*)

have *minus-commute*: $-a - b = -b - a$ **for** *a b* :: *complex* **by** (*simp add*: *algebra-simps*)
define *I* **where** $I = (\lambda n.\ (\textit{of-nat}\ n + a)\ \textit{powr}\ (1 - s) / (1 - s))$
define *R* **where** $R = (\lambda n.\ \textit{EM-remainder}'\ 1\ (\textit{?fs}\ 1)\ (\textit{real}\ 0)\ (\textit{real}\ n))$
define *R-lim* **where** $\textit{R-lim} = \textit{EM-remainder}\ 1\ (\textit{?fs}\ 1)\ 0$
define *C* **where** $C = -\ (a\ \textit{powr}\ -s / 2)$
define *D* **where** $D = (\lambda n.\ (1/2) * (\textit{of-real}\ (a + \textit{real}\ n)\ \textit{powr} - s))$
define D' **where** $D' = (\lambda n.\ \textit{of-real}\ (a + \textit{real}\ n)\ \textit{powr} - s)$
define C' **where** $C' = a\ \textit{powr}\ (1 - s) / (1 - s)$
define C'' **where** $C'' = \textit{of-real}\ a\ \textit{powr} - s$
{
fix *n* :: *nat* **assume** *n*: $n > 0$
have $((\lambda x.\ \textit{of-real}\ (x + a)\ \textit{powr}\ -s)$ *has-integral* $(\textit{of-real}\ (\textit{real}\ n + a)\ \textit{powr}$
$(1-s) / (1 - s) -$
$\textit{of-real}\ (0 + a)\ \textit{powr}\ (1 - s) / (1 - s)))\ \{0..\textit{real}\ n\}$ **using** *n assms*
by (*intro fundamental-theorem-of-calculus*)
(*auto intro*!: *continuous-intros has-vector-derivative-real-field derivative-eq-intros*
simp: *complex-nonpos-Reals-iff*)
hence *I*: $((\lambda x.\ \textit{of-real}\ (x + a)\ \textit{powr}\ -s)$ *has-integral* $(I\ n - C'))\ \{0..n\}$
by (*auto simp*: *divide-simps* C'*-def I-def*)
have $(\sum i\in\{0<..n\}.\ \textit{?f}\ (\textit{real}\ i)) - \textit{integral}\ \{\textit{real}\ 0..\textit{real}\ n\}\ \textit{?f} =$
$(\sum k<1.\ (\textit{bernoulli}'\ (\textit{Suc}\ k) / \textit{fact}\ (\textit{Suc}\ k)) *_R (\textit{?fs}\ k\ (\textit{real}\ n) - \textit{?fs}\ k$
$(\textit{real}\ 0))) + R\ n$
using *n assms* **unfolding** *R-def*
by (*intro euler-maclaurin-strong-raw-nat*[**where** $Y = \{0\}$])
(*auto intro*!: *continuous-intros derivative-eq-intros has-vector-derivative-real-field*
simp: *pochhammer-rec*$'$ *algebra-simps complex-nonpos-Reals-iff*
add-eq-0-iff)
also have $(\sum k<1.\ (\textit{bernoulli}'\ (\textit{Suc}\ k) / \textit{fact}\ (\textit{Suc}\ k)) *_R (\textit{?fs}\ k\ (\textit{real}\ n) - \textit{?fs}$
$k\ (\textit{real}\ 0))) =$
$((n + a)\ \textit{powr} - s - a\ \textit{powr} - s) / 2$
by (*simp add*: *lessThan-nat-numeral scaleR-conv-of-real numeral-2-eq-2* [*symmetric*])
also have $\ldots = C + D\ n$ **by** (*simp add*: *C-def D-def field-simps*)
also have $\textit{integral}\ \{\textit{real}\ 0..\textit{real}\ n\}\ (\lambda x.\ \textit{complex-of-real}\ (x + a)\ \textit{powr} - s) = I$
$n - C'$
using *I* **by** (*simp add*: *has-integral-iff*)
also have $(\sum i\in\{0<..n\}.\ \textit{of-real}\ (\textit{real}\ i + a)\ \textit{powr} - s) =$
$(\sum i=0..n.\ \textit{of-real}\ (\textit{real}\ i + a)\ \textit{powr} - s) - \textit{of-real}\ a\ \textit{powr}\ -s$
using *assms* **by** (*subst sum.head*) *auto*
also have $(\sum i=0..n.\ \textit{of-real}\ (\textit{real}\ i + a)\ \textit{powr} - s) = S\ n + \textit{of-real}\ (\textit{real}\ n +$
$a)\ \textit{powr}\ -s$
unfolding *S-def* **by** (*subst sum.last-plus*) (*auto simp*: *atLeast0LessThan*)
finally have $C - C' + C'' - D'\ n + D\ n + R\ n + (I\ n - I'\ n) = S\ n - I'\ n$
by (*simp add*: *algebra-simps S-def* D'*-def* C''*-def*)
}
hence *ev*: *eventually* $(\lambda n.\ C - C' + C'' - D'\ n + D\ n + R\ n + (I\ n - I'\ n)$
$= S\ n - I'\ n)$ *at-top*
by (*intro eventually-mono*[*OF eventually-gt-at-top*[*of 0*]]) *auto*

have [*simp*]: $-1 - s = -s - 1$ **by** *simp*
{
let *?C* = *norm* (*pochhammer s 1*)
have $R \longrightarrow$ *R-lim* **unfolding** *R-def R-lim-def of-nat-0*
proof (*subst of-int-0* [*symmetric*], *rule tendsto-EM-remainder*)
show *eventually* (λx. *norm* (*?fs 1 x*) $\leq$ *?C* $*$ ($x + a$) *powr* ($-$*Re s* $- 1$)) *at-top*
using *eventually-ge-at-top*[*of 0*]
by *eventually-elim* (*insert assms*, *auto simp*: *norm-mult norm-powr-real-powr*)
next
fix *x* **assume** *x*: $x \geq$ *real-of-int 0*
have [*simp*]: $-$*numeral n* $-$ (x :: *real*) $= -x -$ *numeral n* **for** *x n* **by** (*simp add*: *algebra-simps*)
show ((λx. *?C* / ($-$*Re s*) $*$ ($x + a$) *powr* ($-$*Re s*)) *has-real-derivative*
?C $*$ ($x + a$) *powr* ($-$ *Re s* $- 1$)) (*at x within* {*real-of-int 0*..})
using *assms x* **by** (*auto intro*!: *derivative-eq-intros*)
next
have (λy. *?C* / ($-$ *Re s*) $*$ (a + *real y*) *powr* ($-$ *Re s*)) $\longrightarrow$ *0*
by (*intro tendsto-mult-right-zero tendsto-neg-powr filterlim-real-sequentially filterlim-tendsto-add-at-top*[*OF tendsto-const*]) (*use assms* **in** *auto*)
thus *convergent* (λy. *?C* / ($-$ *Re s*) $*$ (*real y* + a) *powr* ($-$ *Re s*))
by (*auto simp*: *add-ac convergent-def*)
qed (*intro integrable-EM-remainder′ continuous-intros*, *insert assms*, *auto simp*: *add-eq-0-iff*)
}
moreover have (λn. *I n* $-$ *I′ n*) $\longrightarrow$ *0*
proof $-$
have (λn. (*complex-of-real* (*real n* + a) *powr* ($1 - s$) $-$
of-real (*real n*) *powr* ($1 - s$)) / ($1 - s$)) $\longrightarrow$ *0* / ($1 - s$)
using *assms*($3-5$) **by** (*intro filterlim-compose*[*OF - filterlim-real-sequentially*]
tendsto-divide complex-powr-add-minus-powr-asymptotics) *auto*
thus (λn. *I n* $-$ *I′ n*) $\longrightarrow$ *0* **by** (*simp add*: *I-def I′-def divide-simps*)
qed
ultimately have (λn. $C - C' + C'' - D'$ *n* + *D n* + *R n* + (*I n* $-$ *I′ n*))
$\longrightarrow C - C' + C'' - 0 + 0 +$ *R-lim* $+ 0$
unfolding *D-def D′-def* **using** *assms*
by (*intro tendsto-add tendsto-diff tendsto-const tendsto-mult-right-zero*
tendsto-neg-powr-complex-of-real filterlim-tendsto-add-at-top
filterlim-real-sequentially) *auto*
also have $C - C' + C'' - 0 + 0 +$ *R-lim* $+ 0 =$
(a *powr* $-$ s / 2) + a *powr* ($1 - s$) / ($s - 1$) + *R-lim*
by (*simp add*: *C-def C′-def C′′-def field-simps*)
also have ... = *hurwitz-zeta a s*
using *assms* **by** (*simp add*: *hurwitz-zeta-def pre-zeta-def pre-zeta-aux-def*
R-lim-def scaleR-conv-of-real)
finally have (λn. $C - C' + C'' - D'$ *n* + *D n* + *R n* + (*I n* $-$ *I′ n*)) $\longrightarrow$ *hurwitz-zeta a s* **.**
with *ev* **show** *?thesis*

by (*blast intro*: *Lim-transform-eventually*)
qed

lemma *zeta-critical-strip*:
fixes *s* :: *complex* **and** *a* :: *real*
defines $S \equiv (\lambda n.\ \sum i{=}1..n.\ (\textit{of-nat}\ i)\ \textit{powr}\ -\ s)$
defines $I \equiv (\lambda n.\ \textit{of-nat}\ n\ \textit{powr}\ (1\ -\ s)\ /\ (1\ -\ s))$
assumes *s*: $\textit{Re}\ s > 0\ s \neq 1$
shows $(\lambda n.\ S\ n\ -\ I\ n) \longrightarrow \textit{zeta}\ s$
proof −
from *hurwitz-zeta-critical-strip*[*OF s zero-less-one*]
have $(\lambda n.\ (\sum i{<}n.\ \textit{complex-of-real}\ (\textit{Suc}\ i)\ \textit{powr}\ -\ s)\ -$
$\textit{of-nat}\ n\ \textit{powr}\ (1\ -\ s)\ /\ (1\ -\ s)) \longrightarrow \textit{hurwitz-zeta}\ 1\ s$ **by** (*simp add*: *add-ac*)
also have $(\lambda n.\ (\sum i{<}n.\ \textit{complex-of-real}\ (\textit{Suc}\ i)\ \textit{powr}\ -s)) = (\lambda n.\ (\sum i{=}1..n.\ \textit{of-nat}\ i\ \textit{powr}\ -s))$
by (*intro ext sum.reindex-bij-witness*[*of* - $\lambda x.\ x\ -\ 1$ *Suc*]) *auto*
finally show *?thesis* **by** (*simp add*: *zeta-def S-def I-def*)
qed

2.4 The non-vanishing of ζ for $\Re(s) \geq 1$

This proof is based on a sketch by Newman [6], which was previously formalised in HOL Light by Harrison [5], albeit in a much more concrete and low-level style.

Our aim here is to reproduce Newman's proof idea cleanly and on the same high level of abstraction.

theorem *zeta-Re-ge-1-nonzero*:
fixes *s* **assumes** $\textit{Re}\ s \geq 1\ s \neq 1$
shows $\textit{zeta}\ s \neq 0$
proof (*cases* $\textit{Re}\ s > 1$)
case *False*
define *a* **where** $a = -\textit{Im}\ s$
from *False assms* **have** *s* [*simp*]: $s = 1\ -\ \mathrm{i} * a$ **and** *a*: $a \neq 0$
by (*auto simp*: *complex-eq-iff a-def*)
show *?thesis*
proof
assume $\textit{zeta}\ s = 0$
hence *zero*: $\textit{zeta}\ (1\ -\ \mathrm{i} * a) = 0$ **by** *simp*
with *zeta-cnj*[*of* $1\ -\ \mathrm{i} * a$] **have** *zero'*: $\textit{zeta}\ (1\ +\ \mathrm{i} * a) = 0$ **by** *simp*

— We define the function $Q(s) = \zeta(s)^2\zeta(s+ia)\zeta(s-ia)$ and its Dirichlet series. The objective will be to show that this function is entire and its Dirichlet series converges everywhere. Of course, $Q(s)$ has singularities at 1 and $1 \pm ia$, so we need to show they can be removed.
define *Q Q-fds*
where $Q = (\lambda s.\ \textit{zeta}\ s\ \hat{}\ 2 * \textit{zeta}\ (s + \mathrm{i} * a) * \textit{zeta}\ (s - \mathrm{i} * a))$
and $\textit{Q-fds} = \textit{fds-zeta}\ \hat{}\ 2 * \textit{fds-shift}\ (\mathrm{i} * a)\ \textit{fds-zeta} * \textit{fds-shift}\ (-\mathrm{i} * a)$

fds-zeta
let *?sings* = {*1*, *1* + i ∗ *a*, *1* − i ∗ *a*}

— We show that Q is locally bounded everywhere. This is the case because the poles of $\zeta(s)$ cancel with the zeros of $\zeta(s \pm ia)$ and vice versa. This boundedness is then enough to show that Q has only removable singularities.
have *Q-bigo-1*: *Q* ∈ *O*[*at s*](λ-. *1*) **for** *s*
proof −
have *Q-eq*: *Q* = (λ*s*. (*zeta s* ∗ *zeta* (*s* + i ∗ *a*)) ∗ (*zeta s* ∗ *zeta* (*s* − i ∗ *a*)))
by (*simp add*: *Q-def power2-eq-square mult-ac*)

— The singularity of $\zeta(s)$ at 1 gets cancelled by the zero of $\zeta(s - ia)$:
have *bigo1*: (λ*s*. *zeta s* ∗ *zeta* (*s* − i ∗ *a*)) ∈ *O*[*at 1*](λ-. *1*)
if *zeta* (*1* − i ∗ *a*) = *0 a* ≠ *0* **for** *a* :: *real*
proof −
have (λ*s*. *zeta* (*s* − i ∗ *a*) − *zeta* (*1* − i ∗ *a*)) ∈ *O*[*at 1*](λ*s*. *s* − *1*)
using *that*
by (*intro taylor-bigo-linear holomorphic-on-imp-differentiable-at*[*of* - −{*1* + i ∗ *a*}]
holomorphic-intros) (*auto simp*: *complex-eq-iff*)
hence (λ*s*. *zeta s* ∗ *zeta* (*s* − i ∗ *a*)) ∈ *O*[*at 1*](λ*s*. *1* / (*s* − *1*) ∗ (*s* − *1*))
using *that* **by** (*intro landau-o.big.mult zeta-bigo-at-1*) *simp-all*
also have (λ*s*. *1* / (*s* − *1*) ∗ (*s* − *1*)) ∈ Θ[*at 1*](λ-. *1*)
by (*intro bigthetaI-cong*) (*auto simp*: *eventually-at-filter*)
finally show *?thesis* **.**
qed

— The analogous result for $\zeta(s)\zeta(s + ia)$:
have *bigo1'*: (λ*s*. *zeta s* ∗ *zeta* (*s* + i ∗ *a*)) ∈ *O*[*at 1*](λ-. *1*)
if *zeta* (*1* − i ∗ *a*) = *0 a* ≠ *0* **for** *a* :: *real*
using *bigo1*[*of* −*a*] *that zeta-cnj*[*of 1* − i ∗ *a*] **by** *simp*

— The singularity of $\zeta(s - ia)$ gets cancelled by the zero of $\zeta(s)$:
have *bigo2*: (λ*s*. *zeta s* ∗ *zeta* (*s* − i ∗ *a*)) ∈ *O*[*at* (*1* + i ∗ *a*)](λ-. *1*)
if *zeta* (*1* − i ∗ *a*) = *0 a* ≠ *0* **for** *a* :: *real*
proof −
have (λ*s*. *zeta s* ∗ *zeta* (*s* − i ∗ *a*)) ∈ *O*[*filtermap* (λ*s*. *s* + i ∗ *a*) (*at 1*)](λ-. *1*)
using *bigo1'*[*of a*] *that* **by** (*simp add*: *mult.commute landau-o.big.in-filtermap-iff*)
also have *filtermap* (λ*s*. *s* + i ∗ *a*) (*at 1*) = *at* (*1* + i ∗ *a*)
using *filtermap-at-shift*[*of* −i ∗ *a 1*] **by** *simp*
finally show *?thesis* **.**
qed

— Again, the analogous result for $\zeta(s)\zeta(s + ia)$:
have *bigo2'*: (λ*s*. *zeta s* ∗ *zeta* (*s* + i ∗ *a*)) ∈ *O*[*at* (*1* − i ∗ *a*)](λ-. *1*)
if *zeta* (*1* − i ∗ *a*) = *0 a* ≠ *0* **for** *a* :: *real*
using *bigo2*[*of* −*a*] *that zeta-cnj*[*of 1* − i ∗ *a*] **by** *simp*

— Now the final case distinction to show $Q(s) \in O(1)$ for all $s \in \mathbb{C}$:

consider *s = 1* | *s = 1* + i ∗ *a* | *s = 1* − i ∗ *a* | *s* ∉ *?sings* **by** *blast*
thus *?thesis*
proof *cases*
  case *1*
  thus *?thesis* **unfolding** *Q-eq* **using** *zero zero′ a*
    by (*auto intro*: *bigo1 bigo1′ landau-o.big.mult-in-1*)
next
  case *2*
  from *a* **have** *isCont* (λ*s. zeta s* ∗ *zeta* (*s* + i ∗ *a*)) (*1* + i ∗ *a*)
    by (*auto intro*!: *continuous-intros*)
  with *2* **show** *?thesis* **unfolding** *Q-eq* **using** *zero zero′ a*
    by (*auto intro*: *bigo2 landau-o.big.mult-in-1 continuous-imp-bigo-1*)
next
  case *3*
  from *a* **have** *isCont* (λ*s. zeta s* ∗ *zeta* (*s* − i ∗ *a*)) (*1* − i ∗ *a*)
    by (*auto intro*!: *continuous-intros*)
  with *3* **show** *?thesis* **unfolding** *Q-eq* **using** *zero zero′ a*
    by (*auto intro*: *bigo2′ landau-o.big.mult-in-1 continuous-imp-bigo-1*)
 qed (*auto intro*!: *continuous-imp-bigo-1 continuous-intros simp*: *Q-def complex-eq-iff*)
qed

— Thus, we can remove the singularities from Q and extend it to an entire function.

have ∃ *Q′*. *Q′ holomorphic-on UNIV* ∧ (∀ *z*∈*UNIV* − *?sings*. *Q′ z* = *Q z*)
  by (*intro removable-singularities Q-bigo-1*)
    (*auto simp*: *Q-def complex-eq-iff intro*!: *holomorphic-intros*)
then obtain *Q′* **where** *Q′*: *Q′ holomorphic-on UNIV* ⋀*z. z* ∉ *?sings* ⟹ *Q′ z* = *Q z* **by** *blast*

— Q' constitutes an analytic continuation of the Dirichlet series of Q.

have *eval-Q-fds*: *eval-fds Q-fds s* = *Q′ s* **if** *Re s* > *1* **for** *s*
proof −
  have *eval-fds Q-fds s* = *Q s* **using** *that*
  by (*simp add*: *Q-fds-def Q-def eval-fds-mult eval-fds-power fds-abs-converges-mult*
          fds-abs-converges-power eval-fds-zeta)
  also from *that* **have** . . . = *Q′ s* **by** (*subst Q′*) *auto*
  finally show *?thesis* **.**
qed

— Since $\zeta(s)$ and $\zeta(s \pm ia)$ are completely multiplicative Dirichlet series, the logarithm of their product can be rewritten into the following nice form:

have *ln-Q-fds-eq*:
  fds-ln 0 Q-fds = *fds* (λ*k. of-real* (*2* ∗ *mangoldt k* / *ln k* ∗ (*1* + *cos* (*a* ∗ *ln k*))))
proof −
  note *simps* = *fds-ln-mult*[**where** *l′* = *0* **and** *l′′* = *0*] *fds-ln-power*[**where** *l′*

= 0]
fds-ln-prod[**where** $l' = \lambda$-. 0]
have fds-ln 0 Q-fds = 2 ∗ fds-ln 0 fds-zeta + fds-shift (i ∗ a) (fds-ln 0 fds-zeta) +
fds-shift (−i ∗ a) (fds-ln 0 fds-zeta)
by (auto simp: Q-fds-def simps)
also have completely-multiplicative-function (fds-nth (fds-zeta :: complex fds))
by standard auto
hence fds-ln (0 :: complex) fds-zeta = fds (λn. mangoldt n $/_R$ ln (real n))
by (subst fds-ln-completely-multiplicative) (auto simp: fds-eq-iff)
also have 2 ∗ . . . + fds-shift (i ∗ a) . . . + fds-shift (−i ∗ a) . . . =
fds (λk. of-real (2 ∗ mangoldt k / ln k ∗ (1 + cos (a ∗ ln k))))
(**is** ?a = ?b)
proof (intro fds-eqI, goal-cases)
case (1 n)
then consider n = 1 | n > 1 **by** force
hence fds-nth ?a n = mangoldt n / ln (real n) ∗ (2 + (n powr (i ∗ a) + n powr (−i ∗ a)))
by cases (auto simp: field-simps scaleR-conv-of-real numeral-fds)
also have n powr (i ∗ a) + n powr (−i ∗ a) = 2 ∗ cos (of-real (a ∗ ln n))
using 1 **by** (subst cos-exp-eq) (simp-all add: powr-def algebra-simps)
also have mangoldt n / ln (real n) ∗ (2 + . . .) =
of-real (2 ∗ mangoldt n / ln n ∗ (1 + cos (a ∗ ln n)))
by (subst cos-of-real) simp-all
finally show ?case **by** (simp add: fds-nth-fds′)
qed
finally show ?thesis **.**
qed
— It is then obvious that this logarithm series has non-negative real coefficients.
also have nonneg-dirichlet-series . . .
proof (standard, goal-cases)
case (1 n)
from cos-ge-minus-one[of a ∗ ln n] **have** 1 + cos (a ∗ ln (real n)) ≥ 0 **by** linarith
thus ?case **using** 1
by (cases n = 0)
(auto simp: complex-nonneg-Reals-iff fds-nth-fds′ mangoldt-nonneg
intro!: divide-nonneg-nonneg mult-nonneg-nonneg)
qed
— Therefore, the original series also has non-negative real coefficients.
finally have nonneg: nonneg-dirichlet-series Q-fds
by (rule nonneg-dirichlet-series-lnD) (auto simp: Q-fds-def)

— By the Pringsheim–Landau theorem, a Dirichlet series with non-negative coefficnets that can be analytically continued to the entire complex plane must converge everywhere, i. e. its abscissa of (absolute) convergence is $-\infty$:
have abscissa-Q-fds: abs-conv-abscissa Q-fds ≤ 1
unfolding Q-fds-def **by** (auto intro!: abs-conv-abscissa-mult-leI abs-conv-abscissa-power-leI)
with nonneg **and** eval-Q-fds **and** ‹Q′ holomorphic-on UNIV›

have *abscissa*: *abs-conv-abscissa Q-fds* = $-\infty$
by (*intro entire-continuation-imp-abs-conv-abscissa-MInfty*[**where** *c* = *1* **and** *g* = *Q'*])
(*auto simp*: *one-ereal-def*)

— This now leads to a contradiction in a very obvious way. If *Q-fds* is absolutely convergent, then the subseries corresponding to powers of 2 (i.e. we delete all summands a_n/n^s where n is not a power of 2 from the sum) is also absolutely convergent. We denote this series with R.

define *R-fds* **where** *R-fds* = *fds-primepow-subseries 2 Q-fds*
have *conv-abscissa R-fds* ≤ *abs-conv-abscissa R-fds* **by** (*rule conv-le-abs-conv-abscissa*)
also have *abs-conv-abscissa R-fds* ≤ *abs-conv-abscissa Q-fds*
unfolding *R-fds-def* **by** (*rule abs-conv-abscissa-restrict*)
also have ... = $-\infty$ **by** (*simp add*: *abscissa*)
finally have *abscissa'*: *conv-abscissa R-fds* = $-\infty$ **by** *simp*

— Since $\zeta(s)$ and $\zeta(s \pm ia)$ have an Euler product expansion for $\Re(s) > 1$, we have

$$R(s) = (1 - 2^{-s})^-2(1 - 2^{-s+ia})^{-1}(1 - 2^{-s-ia})^{-1}$$

there, and since R converges everywhere and the right-hand side is holomorphic for $\Re(s) > 0$, the equation is also valid for all s with $\Re(s) > 0$ by analytic continuation.

have *eval-R*: *eval-fds R-fds s* =
1 / ((*1* − *2 powr* −*s*) ^ *2* ∗ (*1* − *2 powr* (−*s* + i ∗ *a*)) ∗ (*1* − *2 powr* (−*s* − i ∗ *a*)))
(**is** - = *?f s*) **if** *Re s* > *0* **for** *s*
proof −
show *?thesis*
proof (*rule analytic-continuation-open*[**where** *f* = *eval-fds R-fds*])
show *?f holomorphic-on* {*s*. *Re s* > *0*}
by (*intro holomorphic-intros*) (*auto simp*: *powr-def exp-eq-1 Ln-Reals-eq*)
next
fix *z* **assume** *z*: *z* ∈ {*s*. *Re s* > *1*}
have [*simp*]: *completely-multiplicative-function* (*fds-nth fds-zeta*) **by** *standard auto*
thus *eval-fds R-fds z* = *?f z* **using** *z*
by (*simp add*: *R-fds-def Q-fds-def eval-fds-mult eval-fds-power fds-abs-converges-mult*
fds-abs-converges-power fds-primepow-subseries-euler-product-cm divide-simps
powr-minus powr-diff powr-add fds-abs-summable-zeta)
qed (*insert that abscissa'*, *auto intro*!: *exI*[*of* - *2*] *convex-connected open-halfspace-Re-gt*
convex-halfspace-Re-gt holomorphic-intros)
qed

— We now clearly have a contradiction: $R(s)$, being entire, is continuous everywhere, while the function on the right-hand side clearly has a pole at 0.

show *False*
proof (*rule not-tendsto-and-filterlim-at-infinity*)
have ((λ*b*. (*1*−*2 powr* − *b*)^2 ∗ (*1* − *2 powr* (−*b*+i∗*a*)) ∗ (*1* − *2 powr*

$(-b-\mathrm{i}*a))) \longrightarrow 0)$
$(at\ 0\ within\ \{s.\ Re\ s > 0\})$
(**is** *filterlim ?f′ - -*) **by** (*intro tendsto-eq-intros*) (*auto*)
moreover have *eventually* ($\lambda s.\ s \in \{s.\ Re\ s > 0\}$) ($at\ 0\ within\ \{s.\ Re\ s > 0\}$)
by (*auto simp*: *eventually-at-filter*)
hence *eventually* ($\lambda s.\ ?f'\ s \neq 0$) ($at\ 0\ within\ \{s.\ Re\ s > 0\}$)
by *eventually-elim* (*auto simp*: *powr-def exp-eq-1 Ln-Reals-eq*)
ultimately have *filterlim ?f′* (*at 0*) ($at\ 0\ within\ \{s.\ Re\ s > 0\}$) **by** (*simp add*: *filterlim-at*)
hence *filterlim ?f at-infinity* ($at\ 0\ within\ \{s.\ Re\ s > 0\}$) (**is** *?lim*)
by (*intro filterlim-divide-at-infinity*[*OF tendsto-const*]
tendsto-mult-filterlim-at-infinity) *auto*
also have *ev*: *eventually* ($\lambda s.\ Re\ s > 0$) ($at\ 0\ within\ \{s.\ Re\ s > 0\}$)
by (*auto simp*: *eventually-at intro*!: *exI*[*of - 1*])
have *?lim* $\longleftrightarrow$ *filterlim* (*eval-fds R-fds*) *at-infinity* ($at\ 0\ within\ \{s.\ Re\ s > 0\}$)
by (*intro filterlim-cong refl eventually-mono*[*OF ev*]) (*auto simp*: *eval-R*)
finally show . . . **.**
next
have *continuous* ($at\ 0\ within\ \{s.\ Re\ s > 0\}$) (*eval-fds R-fds*)
by (*intro continuous-intros*) (*auto simp*: *abscissa′*)
thus ((*eval-fds R-fds* $\longrightarrow$ *eval-fds R-fds 0*)) ($at\ 0\ within\ \{s.\ Re\ s > 0\}$)
by (*auto simp*: *continuous-within*)
next
have $0 \in \{s.\ Re\ s \geq 0\}$ **by** *simp*
also have $\{s.\ Re\ s \geq 0\} = closure\ \{s.\ Re\ s > 0\}$
using *closure-halfspace-gt*[*of 1::complex 0*] **by** (*simp add*: *inner-commute*)
finally have $0 \in$ . . . **.**
thus $at\ 0\ within\ \{s.\ Re\ s > 0\} \neq bot$
by (*subst at-within-eq-bot-iff*) *auto*
qed
qed
qed (*fact zeta-Re-gt-1-nonzero*)

2.5 Special values of the ζ functions

theorem *hurwitz-zeta-neg-of-nat*:
assumes $a > 0$
shows *hurwitz-zeta a* ($-$*of-nat n*) = $-$*bernpoly* (*Suc n*) *a* / *of-nat* (*Suc n*)
proof $-$
have $-$*of-nat n* $\neq$ (*1::complex*) **by** (*simp add*: *complex-eq-iff*)
hence *hurwitz-zeta a* ($-$*of-nat n*) =
pre-zeta a ($-$*of-nat n*) + *a powr real* (*Suc n*) / ($-$*of-nat* (*Suc n*))
unfolding *zeta-def hurwitz-zeta-def* **using** *assms* **by** (*simp add*: *powr-of-real* [*symmetric*])
also have *a powr real* (*Suc n*) / ($-$*of-nat* (*Suc n*)) = $-$ (*a powr real* (*Suc n*) / *of-nat* (*Suc n*))
by (*simp add*: *divide-simps del*: *of-nat-Suc*)

also have *a powr real* (*Suc n*) = *a* ^ *Suc n*
using *assms* **by** (*intro powr-realpow*)
also have *pre-zeta a* (−*of-nat n*) = *pre-zeta-aux* (*Suc n*) *a* (− *of-nat n*)
using *assms* **by** (*intro pre-zeta-aux-eq-pre-zeta* [*symmetric*]) *auto*
also have ... = *of-real a powr of-nat n* / *2* +
($\sum$ *i* = *1*..*Suc n*. (*bernoulli* (*2* ∗ *i*) / *fact* (*2* ∗ *i*)) $*_R$
(*pochhammer* (− *of-nat n*) (*2* ∗ *i* − *1*) ∗
of-real a powr (*of-nat n* − *of-nat* (*2* ∗ *i* − *1*)))) +
EM-remainder (*Suc* (*2* ∗ *Suc n*)) (λ*x*. − (*pochhammer* (− *of-nat* *n*)
(*2*∗*n*+*3*) ∗ *of-real* (*x* + *a*) *powr* (− *of-nat n* − *3*))) *0*
(**is** - = *?B* + *sum* (λ*n*. *?f* (*2* ∗ *n*)) - + -)
unfolding *pre-zeta-aux-def* **by** (*simp add*: *add-ac eval-nat-numeral*)
also have *?B* = *of-real* (*a* ^ *n*) / *2*
using *assms* **by** (*subst powr-Reals-eq*) (*auto simp*: *powr-realpow*)
also have *pochhammer* (−*of-nat n* :: *complex*) (*2*∗*n*+*3*) = *0*
by (*subst pochhammer-eq-0-iff*) *auto*
finally have *hurwitz-zeta a* (−*of-nat n*) =
− (*a* ^ *Suc n* / *of-nat* (*Suc n*)) + (*a* ^ *n* / *2* + *sum* (λ*n*. *?f* (*2* ∗ *n*)) {*1*..*Suc n*})
by *simp*

also have *sum* (λ*n*. *?f* (*2* ∗ *n*)) {*1*..*Suc n*} = *sum ?f* ((∗) *2* ‘ {*1*..*Suc n*})
by (*intro sum.reindex-bij-witness*[*of* - λ*i*. *i div 2* λ*i*. *2*∗*i*]) *auto*
also have ... = ($\sum$ *i*=*2*..*2*∗*n*+*2*. *?f i*)
proof (*intro sum.mono-neutral-left ballI*, *goal-cases*)
case (*3 i*)
hence *odd i i* ≠ *1* **by** (*auto elim*!: *evenE*)
thus *?case* **by** (*simp add*: *bernoulli-odd-eq-0*)
qed *auto*
also have ... = ($\sum$ *i*=*2*..*Suc n*. *?f i*)
proof (*intro sum.mono-neutral-right ballI*, *goal-cases*)
case (*3 i*)
hence *pochhammer* (−*of-nat n* :: *complex*) (*i* − *1*) = *0*
by (*subst pochhammer-eq-0-iff*) *auto*
thus *?case* **by** *simp*
qed *auto*
also have ... = ($\sum$ *i*=*Suc 1*..*Suc n*. −*of-real* (*real* (*Suc n choose i*) ∗ *bernoulli* *i* ∗
a ^ (*Suc n* − *i*)) / *of-nat* (*Suc n*))
(**is** *sum ?lhs* - = *sum ?f* -)
proof (*intro sum.cong*, *goal-cases*)
case (*2 i*)
hence *of-nat n* − *of-nat* (*i* − *1*) = (*of-nat* (*Suc n* − *i*) :: *complex*)
by (*auto simp*: *of-nat-diff*)
also have *of-real a powr* ... = *of-real* (*a* ^ (*Suc n* − *i*))
using *2 assms* **by** (*subst powr-nat*) *auto*
finally have *A*: *of-real a powr* (*of-nat n* − *of-nat* (*i* − *1*)) = ...
have *pochhammer* (−*of-nat n*) (*i* − *1*) = *complex-of-real* (*pochhammer* (−*real*

n) (*i* − *1*))
  by (*simp add*: *pochhammer-of-real* [*symmetric*])
  also have *pochhammer* (−*real n*) (*i* − *1*) = *pochhammer* (−*of-nat* (*Suc n*)) *i* / (−*1* − *real n*)
  using *2* **by** (*subst* (*2*) *pochhammer-rec-if*) *auto*
  also have −*1* − *real n* = −*real* (*Suc n*) **by** *simp*
  finally have *B*: *pochhammer* (−*of-nat n*) (*i* − *1*) =
  −*complex-of-real* (*pochhammer* (−*real* (*Suc n*)) *i* / *real* (*Suc n*))
  by (*simp del*: *of-nat-Suc*)
  have *?lhs i* = −*complex-of-real* (*bernoulli i* ∗ *pochhammer* (−*real* (*Suc n*)) *i* / *fact i* ∗
  a ^ (*Suc n* − *i*)) / *of-nat* (*Suc n*)
  by (*simp only*: *A B*) (*simp add*: *scaleR-conv-of-real*)
  also have *bernoulli i* ∗ *pochhammer* (−*real* (*Suc n*)) *i* / *fact i* =
  (*real* (*Suc n*) *gchoose i*) ∗ *bernoulli i* **using** *2*
  by (*subst gbinomial-pochhammer*) (*auto simp*: *minus-one-power-iff bernoulli-odd-eq-0*)
  also have *real* (*Suc n*) *gchoose i* = *Suc n choose i*
  by (*subst binomial-gbinomial*) *auto*
  finally show *?case* **by** *simp*
  qed *auto*
  also have *a* ^ *n* / *2* + *sum ?f* {*Suc 1*..*Suc n*} = *sum ?f* {*1*..*Suc n*}
  by (*subst* (*2*) *sum.atLeast-Suc-atMost*) (*simp-all add*: *scaleR-conv-of-real del*: *of-nat-Suc*)
  also have −(*a* ^ *Suc n* / *of-nat* (*Suc n*)) + *sum ?f* {*1*..*Suc n*} = *sum ?f* {*0*..*Suc n*}
  by (*subst* (*2*) *sum.atLeast-Suc-atMost*) (*simp-all add*: *scaleR-conv-of-real*)
  also have ... = − *bernpoly* (*Suc n*) *a* / *of-nat* (*Suc n*)
  unfolding *sum-negf sum-divide-distrib* [*symmetric*] **by** (*simp add*: *bernpoly-def atLeast0AtMost*)
  finally show *?thesis* **.**
qed

lemma *hurwitz-zeta-0* [*simp*]: *a* > *0* ⟹ *hurwitz-zeta a 0* = *1* / *2* − *a*
  using *hurwitz-zeta-neg-of-nat*[*of a 0*] **by** (*simp add*: *bernpoly-def*)

lemma *zeta-0* [*simp*]: *zeta 0* = −*1* / *2*
  by (*simp add*: *zeta-def*)

theorem *zeta-neg-of-nat*:
  zeta (−*of-nat n*) = −*of-real* (*bernoulli′* (*Suc n*)) / *of-nat* (*Suc n*)
  unfolding *zeta-def* **by** (*simp add*: *hurwitz-zeta-neg-of-nat bernpoly-1′*)

corollary *zeta-trivial-zero*:
  assumes *even n n* ≠ *0*
  shows *zeta* (−*of-nat n*) = *0*
  using *zeta-neg-of-nat*[*of n*] *assms* **by** (*simp add*: *bernoulli′-odd-eq-0*)

theorem *zeta-even-nat*:
  zeta (*2* ∗ *of-nat n*) =

of-real $((-1)$ ^ *Suc n* $*$ *bernoulli* $(2 * n) * (2 * pi)$ ^ $(2 * n)$ / $(2 *$ *fact* $(2 * n)))$
proof (*cases n = 0*)
case *False*
hence (λk. *1* / *of-nat* (*Suc k*) ^ (*2* $*$ *n*)) *sums zeta* (*of-nat* (*2* $*$ *n*))
by (*intro sums-zeta-nat*) *auto*
from *sums-unique2* [*OF this nat-even-power-sums-complex*] *False* **show** *?thesis* **by** *simp*
qed (*insert zeta-neg-of-nat*[*of 0*], *simp-all*)

corollary *zeta-even-numeral*:
zeta (*numeral* (*Num.Bit0 n*)) = *of-real*
((− *1*) ^ *Suc* (*numeral n*) $*$ *bernoulli* (*numeral* (*num.Bit0 n*)) $*$
(*2* $*$ *pi*) ^ *numeral* (*num.Bit0 n*) / (*2* $*$ *fact* (*numeral* (*num.Bit0 n*)))) (**is** - = *?rhs*)
proof −
have $*$: (*2* $*$ *numeral n* :: *nat*) = *numeral* (*Num.Bit0 n*)
by (*subst numeral.numeral-Bit0*, *subst mult-2*, *rule refl*)
have *numeral* (*Num.Bit0 n*) = (*2* $*$ *numeral n* :: *complex*)
by (*subst numeral.numeral-Bit0*, *subst mult-2*, *rule refl*)
also have . . . = *2* $*$ *of-nat* (*numeral n*) **by** (*simp only*: *of-nat-numeral of-nat-mult*)
also have *zeta* . . . = *?rhs* **by** (*subst zeta-even-nat*) (*simp only*: $*$)
finally show *?thesis* **.**
qed

corollary *zeta-neg-even-numeral* [*simp*]: *zeta* (−*numeral* (*Num.Bit0 n*)) = *0*
proof −
have −*numeral* (*Num.Bit0 n*) = (−*of-nat* (*numeral* (*Num.Bit0 n*)) :: *complex*)
by *simp*
also have *zeta* . . . = *0*
proof (*rule zeta-trivial-zero*)
have *numeral* (*Num.Bit0 n*) = (*2* $*$ *numeral n* :: *nat*)
by (*subst numeral.numeral-Bit0*, *subst mult-2*, *rule refl*)
also have *even* . . . **by** (*rule dvd-triv-left*)
finally show *even* (*numeral* (*Num.Bit0 n*) :: *nat*) **.**
qed *auto*
finally show *?thesis* **.**
qed

corollary *zeta-neg-numeral*:
zeta (−*numeral n*) =
−*of-real* (*bernoulli′* (*numeral* (*Num.inc n*)) / *numeral* (*Num.inc n*))
proof −
have −*numeral n* = (− *of-nat* (*numeral n*) :: *complex*)
by *simp*
also have *zeta* . . . = −*of-real* (*bernoulli′* (*numeral* (*Num.inc n*)) / *numeral* (*Num.inc n*))
by (*subst zeta-neg-of-nat*) (*simp add*: *numeral-inc*)
finally show *?thesis* **.**

qed

corollary *zeta-neg1*: *zeta* $(-1) = -1 / 12$
using *zeta-neg-of-nat*[*of 1*] **by** (*simp add*: *eval-bernoulli*)

corollary *zeta-neg3*: *zeta* $(-3) = 1 / 120$
by (*simp add*: *zeta-neg-numeral*)

corollary *zeta-neg5*: *zeta* $(-5) = -1 / 252$
by (*simp add*: *zeta-neg-numeral*)

corollary *zeta-2*: *zeta 2* = *pi* ^ *2* / *6*
by (*simp add*: *zeta-even-numeral*)

corollary *zeta-4*: *zeta 4* = *pi* ^ *4* / *90*
by (*simp add*: *zeta-even-numeral fact-num-eq-if*)

corollary *zeta-6*: *zeta 6* = *pi* ^ *6* / *945*
by (*simp add*: *zeta-even-numeral fact-num-eq-if*)

corollary *zeta-8*: *zeta 8* = *pi* ^ *8* / *9450*
by (*simp add*: *zeta-even-numeral fact-num-eq-if*)

2.6 Integral relation between Γ and ζ function

lemma
assumes *z*: *Re z* > *0* **and** *a*: *a* > *0*
shows *Gamma-hurwitz-zeta-aux-integral*:
Gamma z / (*of-nat n* + *of-real a*) *powr z* =
($\int$ *s*∈{*0*<..}. (*s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*)) ∂*lebesgue*)
and *Gamma-hurwitz-zeta-aux-integrable*:
set-integrable lebesgue {*0*<..} (λ*s*. *s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*))
proof −
note *integrable* = *absolutely-integrable-Gamma-integral′* [*OF z*]
let *?INT* = *set-lebesgue-integral lebesgue* {*0*<..} :: (*real* ⇒ *complex*) ⇒ *complex*
let *?INT′* = *set-lebesgue-integral lebesgue* {*0*<..} :: (*real* ⇒ *real*) ⇒ *real*

have *meas1*: *set-borel-measurable lebesgue* {*0*<..}
(λ*x*. *of-real x powr* (*z* − *1*) / *of-real* (*exp* ((*n*+*a*) ∗ *x*)))
unfolding *set-borel-measurable-def*
by (*intro measurable-completion*, *subst measurable-lborel2*,
intro borel-measurable-continuous-on-indicator) (*auto intro*!: *continuous-intros*)
show *integrable1*: *set-integrable lebesgue* {*0*<..}
(λ*s*. *s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*))
using *assms* **by** (*intro absolutely-integrable-Gamma-integral*) *auto*
from *assms* **have** *pos*: *0* < *real n* + *a* **by** (*simp add*: *add-nonneg-pos*)
hence *complex-of-real 0* ≠ *of-real* (*real n* + *a*) **by** (*simp only*: *of-real-eq-iff*)
also have *complex-of-real* (*real n* + *a*) = *of-nat n* + *of-real a* **by** *simp*
finally have *nz*: … ≠ *0* **by** *auto*

have ((λ*t*. *complex-of-real t powr* (*z* − *1*) / *of-real* (*exp t*)) *has-integral Gamma z*) {*0*<..}
  by (*rule Gamma-integral-complex′*) *fact*+
hence *Gamma z* = *?INT* (λ*t*. *t powr* (*z* − *1*) / *of-real* (*exp t*))
  using *set-lebesgue-integral-eq-integral*(*2*) [*OF integrable*]
  by (*simp add*: *has-integral-iff exp-of-real*)
also have *lebesgue* = *density* (*distr lebesgue lebesgue* (λ*t*. (*real n*+*a*) ∗ *t*))
  (λ*x*. *ennreal* (*real n*+*a*))
  using *lebesgue-real-scale*[*of real n* + *a*] *pos* **by** *auto*
also have *set-lebesgue-integral* ... {*0*<..} (λ*t*. *of-real t powr* (*z* − *1*) / *of-real* (*exp t*)) =
  set-lebesgue-integral (*distr lebesgue lebesgue* (λ*t*. (*real n* + *a*) ∗ *t*)) {*0*<..}
  (λ*t*. (*real n* + *a*) ∗ *t powr* (*z* − *1*) / *exp t*) **using** *integrable pos*
  unfolding *set-lebesgue-integral-def*
  by (*subst integral-density*) (*simp-all add*: *exp-of-real algebra-simps scaleR-conv-of-real set-integrable-def*)
also have ... = *?INT* (λ*s*. (*n* + *a*) ∗ (*of-real* (*n*+*a*) ∗ *of-real s*) *powr* (*z* − *1*) /
  of-real (*exp* ((*n*+*a*) ∗ *s*)))
  unfolding *set-lebesgue-integral-def*
proof (*subst integral-distr*)
  show (∗) (*real n* + *a*) ∈ *lebesgue* $\rightarrow_M$ *lebesgue*
    using *lebesgue-measurable-scaling*[*of real n* + *a*, **where** *?′a* = *real*]
    unfolding *real-scaleR-def* .
next
  have (λ*x*. (*n*+*a*) ∗ (*indicator* {*0*<..} *x* $*_R$ (*of-real x powr* (*z* − *1*) / *of-real* (*exp x*))))
    ∈ *lebesgue* $\rightarrow_M$ *borel*
  using *integrable* **unfolding** *set-integrable-def* **by** (*intro borel-measurable-times*) *simp-all*
  thus (λ*x*. *indicator* {*0*<..} *x* $*_R$
    (*complex-of-real* (*real n* + *a*) ∗ *complex-of-real x powr* (*z* − *1*) / *exp x*))
  ∈ *borel-measurable lebesgue* **by** *simp*
qed (*intro Bochner-Integration.integral-cong refl*, *insert pos*,
  auto simp: *indicator-def zero-less-mult-iff*)
also have ... = *?INT* (λ*s*. ((*n*+*a*) *powr z*) ∗ (*s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*))) **using** *pos*
  by (*intro set-lebesgue-integral-cong refl allI impI*, *simp*, *subst powr-times-real*)
  (*auto simp*: *powr-diff*)
also have ... = (*n* + *a*) *powr z* ∗ *?INT* (λ*s*. *s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*))
  unfolding *set-lebesgue-integral-def*
  by (*subst integral-mult-right-zero* [*symmetric*]) *simp-all*
finally show *Gamma z* / (*of-nat n* + *of-real a*) *powr z* =
  ?INT (λ*s*. *s powr* (*z* − *1*) / *exp* ((*n*+*a*) ∗ *s*))
  using *nz* **by** (*auto simp add*: *field-simps*)
qed

lemma
  assumes *x*: $x > 0$ **and** $a > 0$
  shows *Gamma-hurwitz-zeta-aux-integral-real*:
    Gamma x / (*real n* + *a*) *powr x* =
      set-lebesgue-integral lebesgue {*0*<..}
      (λ*s*. *s powr* (*x* − *1*) / *exp* ((*real n* + *a*) ∗ *s*))
  and *Gamma-hurwitz-zeta-aux-integrable-real*:
    set-integrable lebesgue {*0*<..} (λ*s*. *s powr* (*x* − *1*) / *exp* ((*real n* + *a*) ∗
s))
proof −
  show *set-integrable lebesgue* {*0*<..} (λ*s*. *s powr* (*x* − *1*) / *exp* ((*real n* + *a*) ∗ *s*))
    using *absolutely-integrable-Gamma-integral*[*of of-real x real n* + *a*]
    unfolding *set-integrable-def*
  proof (*rule Bochner-Integration.integrable-bound*, *goal-cases*)
    case *3*
    have *set-integrable lebesgue* {*0*<..} (λ*xa*. *complex-of-real xa powr* (*of-real x* −
1) /
      of-real (*exp* ((*n* + *a*) ∗ *xa*)))
      using *assms* **by** (*intro Gamma-hurwitz-zeta-aux-integrable*) *auto*
    also have *?this* ⟷ *integrable lebesgue*
      (λ*s*. *complex-of-real* (*indicator* {*0*<..} *s* $*_R$ (*s powr* (*x* − *1*) / (*exp*
((*n*+*a*) ∗ *s*)))))
      unfolding *set-integrable-def*
      by (*intro Bochner-Integration.integrable-cong refl*) (*auto simp*: *powr-Reals-eq*
indicator-def)
    finally have *set-integrable lebesgue* {*0*<..} (λ*s*. *s powr* (*x* − *1*) / *exp* ((*n*+*a*)
∗ *s*))
      unfolding *set-integrable-def complex-of-real-integrable-eq* **.**
    thus *?case*
      by (*simp add*: *set-integrable-def*)
  qed (*insert assms*, *auto intro*!: *AE-I2 simp*: *indicator-def norm-divide norm-powr-real-powr*)
  from *Gamma-hurwitz-zeta-aux-integral*[*of of-real x a n*] **and** *assms*
    have *of-real* (*Gamma x* / (*real n* + *a*) *powr x*) = *set-lebesgue-integral lebesgue*
{*0*<..}
      (λ*s*. *complex-of-real s powr* (*of-real x* − *1*) / *of-real* (*exp* ((*n*+*a*) ∗ *s*)))
(**is** - = *?I*)
      by (*auto simp*: *Gamma-complex-of-real powr-Reals-eq*)
  also have *?I* = *lebesgue-integral lebesgue*
      (λ*s*. *of-real* (*indicator* {*0*<..} *s* $*_R$ (*s powr* (*x* − *1*) / *exp* ((*n*+*a*)
∗ *s*))))
    unfolding *set-lebesgue-integral-def*
    using *assms* **by** (*intro Bochner-Integration.integral-cong refl*)
      (*auto simp*: *indicator-def powr-Reals-eq*)
  also have ... = *of-real* (*set-lebesgue-integral lebesgue* {*0*<..}
      (λ*s*. *s powr* (*x* − *1*) / *exp* ((*n*+*a*) ∗ *s*)))
    unfolding *set-lebesgue-integral-def*
    by (*rule Bochner-Integration.integral-complex-of-real*)
  finally show *Gamma x* / (*real n* + *a*) *powr x* = *set-lebesgue-integral lebesgue*
{*0*<..}

($\lambda s.$ *s powr* $(x - 1)$ / *exp* ((*real n* + a) $*$ s))
by (*subst* (*asm*) *of-real-eq-iff*)
qed

theorem
assumes *Re z* > *1* **and** $a > (0::real)$
shows *Gamma-times-hurwitz-zeta-integral*: *Gamma z* $*$ *hurwitz-zeta a z* =
($\int x \in \{0<..\}$. (*of-real x powr* $(z - 1)$ $*$ *of-real* (*exp* ($-a*x$) / ($1 -$ *exp* ($-x$)))) ∂*lebesgue*)
and *Gamma-times-hurwitz-zeta-integrable*:
set-integrable lebesgue {*0*<..}
($\lambda x.$ *of-real x powr* $(z - 1)$ $*$ *of-real* (*exp* ($-a*x$) / ($1 -$ *exp* ($-x$))))
proof −
from *assms* **have** *z*: *Re z* > *0* **by** *simp*
let *?INT* = *set-lebesgue-integral lebesgue* {*0*<..} :: (*real* ⇒ *complex*) ⇒ *complex*
let *?INT'* = *set-lebesgue-integral lebesgue* {*0*<..} :: (*real* ⇒ *real*) ⇒ *real*

have *1*: *complex-set-integrable lebesgue* {*0*<..}
($\lambda x.$ *of-real x powr* $(z - 1)$ / *of-real* (*exp* ((*real n* + a) $*$ x))) **for** *n*
by (*rule Gamma-hurwitz-zeta-aux-integrable*) (*use assms* **in** *simp-all*)
have *2*: *summable* ($\lambda n.$ *norm* (*indicator* {*0*<..} s $*_R$ (*of-real s powr* $(z - 1)$ /
of-real (*exp* ((n + a) $*$ s))))) **for** *s*
proof (*cases* $s > 0$)
case *True*
hence *summable* ($\lambda n.$ *norm* (*of-real s powr* $(z - 1)$) $*$ *exp* ($-a * s$) $*$ *exp* ($-s$) ^ n)
using *assms* **by** (*intro summable-mult summable-geometric*) *simp-all*
with *True* **show** *?thesis*
by (*simp add*: *norm-mult norm-divide exp-add exp-diff*
exp-minus field-simps exp-of-nat-mult [*symmetric*])
qed *simp-all*
have *3*: *summable* ($\lambda n.$ $\int x.$ *norm* (*indicator* {*0*<..} x $*_R$ (*complex-of-real x powr* $(z - 1)$ /
complex-of-real (*exp* ((n + a) $*$ x)))) ∂*lebesgue*)
proof −
have *summable* ($\lambda n.$ *Gamma* (*Re z*) $*$ (*real n* + a) *powr* −*Re z*)
using *assms* **by** (*intro summable-mult summable-hurwitz-zeta-real*) *simp-all*
also have *?this* ⟷ *summable* ($\lambda n.$ *?INT'* ($\lambda s.$ *norm* (*of-real s powr* $(z - 1)$ /
of-real (*exp* ((n+a) $*$ s)))))
proof (*intro summable-cong always-eventually allI*, *goal-cases*)
case (*1 n*)
have *Gamma* (*Re z*) $*$ (*real n* + a) *powr* −*Re z* = *Gamma* (*Re z*) / (*real n* + a) *powr Re z*
by (*subst powr-minus*) (*simp-all add*: *field-simps*)
also from *assms* **have** ... = ($\int x \in \{0<..\}$. (x *powr* (*Re z*−*1*) / *exp* ((n+a) $*$ x)) ∂*lebesgue*)
by (*subst Gamma-hurwitz-zeta-aux-integral-real*) *simp-all*
also have ... = ($\int xa \in \{0<..\}$. *norm* (*of-real xa powr* (z−*1*) / *of-real* (*exp*

$((n+a) * xa)))$
$\partial lebesgue)$
unfolding *set-lebesgue-integral-def*
by (*intro Bochner-Integration.integral-cong refl*)
(*auto simp*: *indicator-def norm-divide norm-powr-real-powr*)
finally show *?case* **.**
qed
finally show *?thesis*
by (*simp add*: *set-lebesgue-integral-def*)
qed

have *sum-eq*: $(\sum n.\ indicator\ \{0<..\}\ s *_R (of\text{-}real\ s\ powr\ (z - 1) / of\text{-}real\ (exp\ ((n+a) * s)))) =$
$indicator\ \{0<..\}\ s *_R (of\text{-}real\ s\ powr\ (z - 1) *$
$of\text{-}real\ (exp\ (-a * s) / (1 - exp\ (-s))))$ **for** *s*
proof $(cases\ s > 0)$
case *True*
hence $(\sum n.\ indicator\ \{0..\}\ s *_R (of\text{-}real\ s\ powr\ (z - 1) / of\text{-}real\ (exp\ ((n+a) * s)))) =$
$(\sum n.\ of\text{-}real\ s\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a * s)) * of\text{-}real\ (exp\ (-s))\ \hat{}\ n)$
by (*intro suminf-cong*)
(*auto simp*: *exp-add exp-minus exp-of-nat-mult* [*symmetric*] *field-simps of-real-exp*)
also have $(\sum n.\ of\text{-}real\ s\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a * s)) * of\text{-}real\ (exp\ (-s))\ \hat{}\ n) =$
$of\text{-}real\ s\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a * s)) * (\sum n.\ of\text{-}real\ (exp\ (-s))\ \hat{}\ n)$
using *True* **by** (*intro suminf-mult summable-geometric*) *simp-all*
also have $(\sum n.\ complex\text{-}of\text{-}real\ (exp\ (-s))\ \hat{}\ n) = 1 / (1 - of\text{-}real\ (exp\ (-s)))$
using *True* **by** (*intro suminf-geometric*) *auto*
also have $of\text{-}real\ s\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a * s)) * \ldots =$
$of\text{-}real\ s\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a * s) / (1 - exp\ (-s)))$ **using** ‹$a > 0$›
by (*auto simp add*: *divide-simps exp-minus*)
finally show *?thesis* **using** *True* **by** *simp*
qed *simp-all*

show *set-integrable lebesgue* $\{0<..\}$
$(\lambda x.\ of\text{-}real\ x\ powr\ (z - 1) * of\text{-}real\ (exp\ (-a*x) / (1 - exp\ (-x))))$
using *1* **unfolding** *sum-eq* [*symmetric*] *set-integrable-def*
by (*intro integrable-suminf*[*OF - AE-I2*] *2 3*)

have $(\lambda n.\ ?INT\ (\lambda s.\ s\ powr\ (z - 1) / exp\ ((n+a) * s)))$ *sums lebesgue-integral lebesgue*
$(\lambda s.\ \sum n.\ indicator\ \{0<..\}\ s *_R (s\ powr\ (z - 1) / exp\ ((n+a) * s)))$ (**is** *?A sums ?B*)
using *1* **unfolding** *set-lebesgue-integral-def set-integrable-def*
by (*rule sums-integral*[*OF - AE-I2*[*OF 2*] *3*])

also have *?A* = (*λn. Gamma z* ∗ (*n* + *a*) *powr* −*z*)
using *assms* **by** (*subst Gamma-hurwitz-zeta-aux-integral* [*symmetric*])
(*simp-all add*: *powr-minus divide-simps*)
also have *?B* = *?INT* (*λs. of-real s powr* (*z* − *1*) ∗ *of-real* (*exp* (−*a* ∗ *s*) / (*1* − *exp* (−*s*))))
unfolding *sum-eq set-lebesgue-integral-def* **..**
finally have (*λn. Gamma z* ∗ (*of-nat n* + *of-real a*) *powr* −*z*) *sums*
?INT (*λx. of-real x powr* (*z* − *1*) ∗ *of-real* (*exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*))))
by *simp*
moreover have (*λn. Gamma z* ∗ (*of-nat n* + *of-real a*) *powr* −*z*) *sums* (*Gamma z* ∗ *hurwitz-zeta a z*)
using *assms* **by** (*intro sums-mult sums-hurwitz-zeta*) *simp-all*
ultimately show *Gamma z* ∗ *hurwitz-zeta a z* =
(∫ *x*∈{*0*<..}. (*of-real x powr* (*z* − *1*) ∗
of-real (*exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*)))) ∂*lebesgue*)
by (*rule sums-unique2* [*symmetric*])
qed

corollary
assumes *Re z* > *1*
shows *Gamma-times-zeta-integral*: *Gamma z* ∗ *zeta z* =
(∫ *x*∈{*0*<..}. (*of-real x powr* (*z* − *1*) / *of-real* (*exp x* − *1*)) ∂*lebesgue*) (**is** *?th1*)
and *Gamma-times-zeta-integrable*:
set-integrable lebesgue {*0*<..}
(*λx. of-real x powr* (*z* − *1*) / *of-real* (*exp x* − *1*)) (**is** *?th2*)
proof −
have ∗: (*λx. indicator* {*0*<..} *x* $*_R$ (*of-real x powr* (*z* − *1*) ∗
of-real (*exp* (−*x*) / (*1* − *exp* (−*x*))))) =
(*λx. indicator* {*0*<..} *x* $*_R$ (*of-real x powr* (*z* − *1*) / *of-real* (*exp x* − *1*)))
by (*intro ext*) (*simp add*: *field-simps exp-minus indicator-def*)
from *Gamma-times-hurwitz-zeta-integral* [*OF assms zero-less-one*] **and** ∗
show *?th1* **by** (*simp add*: *zeta-def set-lebesgue-integral-def*)
from *Gamma-times-hurwitz-zeta-integrable* [*OF assms zero-less-one*] **and** ∗
show *?th2* **by** (*simp add*: *zeta-def set-integrable-def*)
qed

corollary *hurwitz-zeta-integral-Gamma-def*:
assumes *Re z* > *1 a* > *0*
shows *hurwitz-zeta a z* =
rGamma z ∗ (∫ *x*∈{*0*<..}. (*of-real x powr* (*z* − *1*) ∗
of-real (*exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*)))) ∂*lebesgue*)
proof −
from *assms* **have** *Gamma z* ≠ *0*
by (*subst Gamma-eq-zero-iff*) (*auto elim*!: *nonpos-Ints-cases*)
with *Gamma-times-hurwitz-zeta-integral* [*OF assms*] **show** *?thesis*
by (*simp add*: *rGamma-inverse-Gamma field-simps*)

qed

corollary *zeta-integral-Gamma-def*:
  assumes *Re z > 1*
  shows *zeta z =*
    rGamma z ∗ (∫ x∈{0<..}. (of-real x powr (z − 1) / of-real (exp x − 1)) ∂lebesgue)
proof −
  from *assms* **have** *Gamma z ≠ 0*
    by (*subst Gamma-eq-zero-iff*) (*auto elim*!: *nonpos-Ints-cases*)
  with *Gamma-times-zeta-integral* [*OF assms*] **show** *?thesis*
    by (*simp add*: *rGamma-inverse-Gamma field-simps*)
qed

lemma *Gamma-times-zeta-has-integral*:
  assumes *Re z > 1*
  shows ((*λx. x powr* (*z* − *1*) / (*of-real* (*exp x*) − *1*)) *has-integral* (*Gamma z* ∗ *zeta z*)) {*0*<..}
    (**is** (*?f has-integral* -) -)
proof −
  have (*?f has-integral set-lebesgue-integral lebesgue* {*0*<..} *?f*) {*0*<..}
    using *Gamma-times-zeta-integrable*[*OF assms*]
    by (*intro has-integral-set-lebesgue*) *auto*
  also have *set-lebesgue-integral lebesgue* {*0*<..} *?f = Gamma z ∗ zeta z*
    using *Gamma-times-zeta-integral*[*OF assms*] **by** *simp*
  finally show *?thesis* **.**
qed

lemma *Gamma-times-zeta-has-integral-real*:
  fixes *z* :: *real*
  assumes *z > 1*
  shows ((*λx. x powr* (*z* − *1*) / (*exp x* − *1*)) *has-integral* (*Gamma z* ∗ *Re* (*zeta z*))) {*0*<..}
proof −
  from *assms* **have** ∗: *Re* (*of-real z*) > *1* **by** *simp*
  have ((*λx. Re* (*complex-of-real x powr* (*complex-of-real z* − *1*)) / (*exp x* − *1*)) *has-integral*
    Gamma z ∗ *Re* (*zeta* (*complex-of-real z*))) {*0*<..}
   using *has-integral-linear*[*OF Gamma-times-zeta-has-integral*[*OF* ∗] *bounded-linear-Re*]
    by (*simp add*: *o-def Gamma-complex-of-real*)
  also have *?this* ⟷ *?thesis*
    using *assms* **by** (*intro has-integral-cong*) (*auto simp*: *powr-Reals-eq*)
  finally show *?thesis* **.**
qed

lemma *Gamma-integral-real*′:
  assumes *x*: *x* > (*0* :: *real*)
  shows ((*λt. t powr* (*x* − *1*) / *exp t*) *has-integral Gamma x*) {*0*<..}

using *Gamma-integral-real*[*OF assms*] **by** (*subst has-integral-closure* [*symmetric*]) *auto*

2.7 An analytic proof of the infinitude of primes

We can now also do an analytic proof of the infinitude of primes.

lemma *primes-infinite-analytic*: *infinite* {*p* :: *nat*. *prime p*}
proof
— Suppose the set of primes were finite.
define *P* :: *nat set* **where** *P* = {*p*. *prime p*}
assume *fin*: *finite P*

— Then the Euler product form of the ζ function ranges over a finite set, and since each factor is holomorphic in the positive real half-space, the product is, too.
define *zeta'* :: *complex* ⇒ *complex*
where *zeta'* = (λ*s*. (∏ *p*∈*P*. *inverse* (*1* − *1* / *of-nat p powr s*)))
have *holo*: *zeta' holomorphic-on A* **if** *A* ⊆ {*s*. *Re s* > *0*} **for** *A*
proof −
{
fix *p* :: *nat* **and** *s* :: *complex* **assume** *p*: *p* ∈ *P* **and** *s*: *s* ∈ *A*
from *p* **have** *p'*: *real p* > *1*
by (*subst of-nat-1* [*symmetric*], *subst of-nat-less-iff*) (*simp add*: *prime-gt-Suc-0-nat P-def*)
have *norm* (*of-nat p powr s*) = *real p powr Re s*
by (*simp add*: *norm-powr-real-powr*)
also have ... > *real p powr 0* **using** *p p' s that*
by (*subst powr-less-cancel-iff*) (*auto simp*: *prime-gt-1-nat*)
finally have *of-nat p powr s* ≠ *1* **using** *p* **by** (*auto simp*: *P-def*)
}
thus *?thesis* **by** (*auto simp*: *zeta'-def P-def intro*!: *holomorphic-intros*)
qed

— Since the Euler product expansion of $\zeta(s)$ is valid for all s with real value at least 1, and both $\zeta(s)$ and the Euler product must be equal in the positive real half-space punctured at 1 by analytic continuation.
have *eq*: *zeta s* = *zeta' s* **if** *Re s* > *0 s* ≠ *1* **for** *s*
proof (*rule analytic-continuation-open*[*of* {*s*. *Re s* > *1*} {*s*. *Re s* > *0*} − {*1*} *zeta zeta'*])
fix *s* **assume** *s*: *s* ∈ {*s*. *Re s* > *1*}
let *?f* = (λ*n*. ∏ *p*≤*n*. *if prime p then inverse* (*1* − *1* / *of-nat p powr s*) *else 1*)
have *eventually* (λ*n*. *?f n* = *zeta' s*) *sequentially*
using *eventually-ge-at-top*[*of Max P*]
proof *eventually-elim*
case (*elim n*)
have *P* ≠ {} **by** (*auto simp*: *P-def intro*!: *exI*[*of* - *2*])
with *elim* **have** *P* ⊆ {..*n*} **using** *fin* **by** *auto*
thus *?case* **unfolding** *zeta'-def*
by (*intro prod.mono-neutral-cong-right*) (*auto simp*: *P-def*)
qed

moreover from *s* **have** *?f* $\longrightarrow$ *zeta s* **by** (*intro euler-product-zeta*) *auto*
ultimately have (λ-. *zeta′ s*) $\longrightarrow$ *zeta s*
by (*blast intro*: *Lim-transform-eventually*)
thus *zeta s* = *zeta′ s* **by** (*simp add*: *LIMSEQ-const-iff*)
qed (*auto intro*!: *exI*[*of - 2*] *open-halfspace-Re-gt connected-open-delete convex-halfspace-Re-gt*
holomorphic-intros holo that intro: *convex-connected*)

— However, since the Euler product is holomorphic on the entire positive real half-space, it cannot have a pole at 1, while $\zeta(s)$ does have a pole at 1. Since they are equal in the punctured neighbourhood of 1, this is a contradiction.
have *ev*: *eventually* (λ*s*. *s* ∈ {*s*. *Re s* > *0*} − {*1*}) (*at 1*)
by (*auto simp*: *eventually-at-filter intro*!: *open-halfspace-Re-gt*
eventually-mono[*OF eventually-nhds-in-open*[*of* {*s*. *Re s* > *0*}]])
have ¬*is-pole zeta′ 1*
by (*rule not-is-pole-holomorphic* [*of* {*s*. *Re s* > *0*}]) (*auto intro*!: *holo open-halfspace-Re-gt*)
also have *is-pole zeta′ 1* $\longleftrightarrow$ *is-pole zeta 1* **unfolding** *is-pole-def*
by (*intro filterlim-cong refl eventually-mono* [*OF ev*] *eq* [*symmetric*]) *auto*
finally show *False* **using** *is-pole-zeta* **by** *contradiction*
qed

2.8 The periodic zeta function

The periodic zeta function $F(s, q)$ (as described e. g. by Apostol [1] is related to the Hurwitz zeta function. It is periodic in q with period 1 and it can be represented by a Dirichlet series that is absolutely convergent for $\Re(s) > 1$. If $q \notin \mathbb{Z}$, it furthermore convergent for $\Re(s) > 0$.

It is clear that for integer q, we have $F(s, q) = F(s, 0) = \zeta(s)$. Moreover, for non-integer q, $F(s, q)$ can be analytically continued to an entire function.

definition *fds-perzeta* :: *real* ⇒ *complex fds* **where**
fds-perzeta q = *fds* (λ*m*. *exp* (*2* ∗ *pi* ∗ i ∗ *m* ∗ *q*))

The definition of the periodic zeta function on the full domain is a bit unwieldy. The precise reasoning for this definition will be given later, and, in any case, it is probably more instructive to look at the derived "alternative" definitions later.

definition *perzeta* :: *real* ⇒ *complex* ⇒ *complex* **where**
perzeta q′ s =
(*if q′* ∈ ℤ *then zeta s*
else let q = *frac q′ in*
if s = *0 then* i / (*2* ∗ *pi*) ∗ (*pre-zeta q 1* − *pre-zeta* (*1* − *q*) *1* +
ln (*1* − *q*) − *ln q* + *pi* ∗ i)
else if s ∈ ℕ *then eval-fds* (*fds-perzeta q*) *s*
else complex-of-real (*2* ∗ *pi*) *powr* (*s* − *1*) ∗ i ∗ *Gamma* (*1* − *s*) ∗
(i *powr* (−*s*) ∗ *hurwitz-zeta q* (*1* − *s*) −
i *powr s* ∗ *hurwitz-zeta* (*1* − *q*) (*1* − *s*)))

interpretation *fds-perzeta*: *periodic-fun-simple′ fds-perzeta*
by *standard* (*simp-all add*: *fds-perzeta-def exp-add ring-distribs exp-eq-polar cis-mult* [*symmetric*] *cis-multiple-2pi*)

interpretation *perzeta*: *periodic-fun-simple′ perzeta*
proof −
have [*simp*]: $n + 1 \in \mathbb{Z} \longleftrightarrow n \in \mathbb{Z}$ **for** *n* :: *real*
by (*simp flip*: *frac-eq-0-iff add*: *frac.plus-1*)
show *periodic-fun-simple′ perzeta*
by *standard* (*auto simp*: *fun-eq-iff perzeta-def Let-def frac.plus-1*)
qed

lemma *perzeta-frac* [*simp*]: *perzeta* (*frac q*) = *perzeta q*
by (*auto simp*: *perzeta-def fun-eq-iff Let-def*)

lemma *fds-perzeta-frac* [*simp*]: *fds-perzeta* (*frac q*) = *fds-perzeta q*
using *fds-perzeta.plus-of-int*[*of frac q* $\lfloor q \rfloor$] **by** (*simp add*: *frac-def*)

lemma *abs-conv-abscissa-perzeta*: *abs-conv-abscissa* (*fds-perzeta q*) $\leq$ *1*
proof (*rule abs-conv-abscissa-leI*)
fix *c* **assume** *c*: *ereal c* > *1*
hence *summable* ($\lambda n.$ *n powr* −*c*)
by (*simp add*: *summable-real-powr-iff*)
also have *?this* $\longleftrightarrow$ *fds-abs-converges* (*fds-perzeta q*) (*of-real c*) **unfolding** *fds-abs-converges-def*
by (*intro summable-cong eventually-mono*[*OF eventually-gt-at-top*[*of 0*]])
(*auto simp*: *norm-divide norm-powr-real-powr fds-perzeta-def powr-minus field-simps*)
finally show $\exists s.\ s \cdot 1 = c \land$ *fds-abs-converges* (*fds-perzeta q*) *s*
by (*intro exI*[*of - of-real c*]) *auto*
qed

lemma *conv-abscissa-perzeta*: *conv-abscissa* (*fds-perzeta q*) $\leq$ *1*
by (*rule order.trans*[*OF conv-le-abs-conv-abscissa abs-conv-abscissa-perzeta*])

lemma *fds-perzeta--left-0* [*simp*]: *fds-perzeta 0* = *fds-zeta*
by (*simp add*: *fds-perzeta-def fds-zeta-def*)

lemma *perzeta-0-left* [*simp*]: *perzeta 0 s* = *zeta s*
by (*simp add*: *perzeta-def eval-fds-zeta*)

lemma *perzeta-int*: $q \in \mathbb{Z} \Longrightarrow$ *perzeta q* = *zeta*
by (*simp add*: *perzeta-def fun-eq-iff*)

lemma *fds-perzeta-int*: $q \in \mathbb{Z} \Longrightarrow$ *fds-perzeta q* = *fds-zeta*
by (*auto simp*: *fds-perzeta.of-int elim*!: *Ints-cases*)

lemma *sums-fds-perzeta*:
assumes *Re s* > *1*

shows $(\lambda m.\ exp\ (2 * pi * \mathrm{i} * Suc\ m * q)\ /\ of\text{-}nat\ (Suc\ m)\ powr\ s)\ sums$
$eval\text{-}fds\ (fds\text{-}perzeta\ q)\ s$
proof −
have *conv-abscissa* (*fds-perzeta q*) $\leq$ *1* **by** (*rule conv-abscissa-perzeta*)
also have ... $<$ *ereal* (*Re s*) **using** *assms* **by** (*simp add*: *one-ereal-def*)
finally have *fds-converges* (*fds-perzeta q*) *s* **by** (*intro fds-converges*) *auto*
hence $(\lambda n.\ fds\text{-}nth\ (fds\text{-}perzeta\ q)\ (Suc\ n)\ /\ nat\text{-}power\ (Suc\ n)\ s)\ sums$
eval-fds (*fds-perzeta q*) *s* **by** (*subst sums-Suc-iff*) (*auto simp*: *fds-converges-iff*)
thus *?thesis* **by** (*simp add*: *fds-perzeta-def*)
qed

lemma *sum-tendsto-fds-perzeta*:
assumes *Re s* $>$ *1*
shows $(\lambda n.\ \sum k\in\{0<..n\}.\ exp\ (2 * real\ k * pi * q * \mathrm{i}) * of\text{-}nat\ k\ powr - s)$
$\longrightarrow eval\text{-}fds\ (fds\text{-}perzeta\ q)\ s$
proof −
have $(\lambda m.\ exp\ (2 * pi * \mathrm{i} * Suc\ m * q)\ /\ of\text{-}nat\ (Suc\ m)\ powr\ s)\ sums\ eval\text{-}fds$
(*fds-perzeta q*) *s*
by (*intro sums-fds-perzeta assms*)
hence $(\lambda n.\ \sum k<n.\ exp\ (2 * real\ (Suc\ k) * pi * q * \mathrm{i}) * of\text{-}nat\ (Suc\ k)\ powr$
$-s) \longrightarrow$
eval-fds (*fds-perzeta q*) *s*
(**is** *filterlim ?f - -*) **by** (*simp add*: *sums-def powr-minus field-simps*)
also have $?f = (\lambda n.\ \sum k\in\{0<..n\}.\ exp\ (2 * real\ k * pi * q * \mathrm{i}) * of\text{-}nat\ k\ powr$
$-s)$
by (*intro ext sum.reindex-bij-betw sum.reindex-bij-witness*[*of - λn. n − 1 Suc*])
auto
finally show *?thesis* **by** *simp*
qed

Using the geometric series, it is easy to see that the Dirichlet series for $F(s, q)$ has bounded partial sums for non-integer q, so it must converge for any s with $\Re(s) > 0$.

lemma *conv-abscissa-perzeta′*:
assumes $q \notin \mathbb{Z}$
shows *conv-abscissa* (*fds-perzeta q*) $\leq$ *0*
proof (*rule conv-abscissa-leI*)
fix *c* :: *real* **assume** *c*: *ereal c* $>$ *0*
have *fds-converges* (*fds-perzeta q*) (*of-real c*)
proof (*rule bounded-partial-sums-imp-fps-converges*)
define ω **where** $\omega = exp\ (2 * pi * \mathrm{i} * q)$
have [*simp*]: *norm* ω = *1* **by** (*simp add*: *ω-def*)
define *M* **where** $M = 2\ /\ norm\ (1 - \omega)$
from ‹$q \notin \mathbb{Z}$› **have** $\omega \neq 1$
by (*auto simp*: *ω-def exp-eq-1*)
hence *M* $>$ *0* **by** (*simp add*: *M-def*)

show $Bseq\ (\lambda n.\ \sum k\leq n.\ fds\text{-}nth\ (fds\text{-}perzeta\ q)\ k\ /\ nat\text{-}power\ k\ 0)$
unfolding *Bseq-def*

```
    proof (rule exI, safe)
      fix n :: nat
      show norm (∑k≤n. fds-nth (fds-perzeta q) k / nat-power k 0) ≤ M
      proof (cases n = 0)
        case False
        have (∑k≤n. fds-nth (fds-perzeta q) k / nat-power k 0) =
                (∑k∈{1..1 + (n − 1)}. ω ^ k) using False
          by (intro sum.mono-neutral-cong-right)
          (auto simp: fds-perzeta-def fds-nth-fds exp-of-nat-mult [symmetric] mult-ac
ω-def)
        also have ... = ω ∗ (1 − ω ^ n) / (1 − ω) using ‹ω ≠ 1› and False
          by (subst sum-gp-offset) simp
        also have norm ... ≤ 1 ∗ (norm (1::complex) + norm (ω ^ n)) / norm (1
− ω)
          unfolding norm-mult norm-divide
          by (intro mult-mono divide-right-mono norm-triangle-ineq4) auto
        also have ... = M by (simp add: M-def norm-power)
        finally show ?thesis .
      qed (use ‹M > 0› in simp-all)
    qed fact+
  qed (insert c, auto)
  thus ∃s. s · 1 = c ∧ fds-converges (fds-perzeta q) s
    by (intro exI[of - of-real c]) auto
qed

lemma fds-perzeta-one-half: fds-perzeta (1 / 2) = fds (λn. (−1) ^ n)
  using Complex.DeMoivre[of pi]
  by (intro fds-eqI) (auto simp: fds-perzeta-def exp-eq-polar mult-ac)

lemma perzeta-one-half-1 [simp]: perzeta (1 / 2) 1 = −ln 2
proof (rule sums-unique2)
  have ∗: (1 / 2 :: real) ∉ ℤ
    using fraction-not-in-ints[of 2 1] by auto
  have fds-converges (fds-perzeta (1 / 2)) 1
    by (rule fds-converges, rule le-less-trans, rule conv-abscissa-perzeta') (use ∗ in
auto)
  hence (λn. (−1) ^ Suc n / Suc n) sums eval-fds (fds-perzeta (1 / 2)) 1
    unfolding fds-converges-altdef by (simp add: fds-perzeta-one-half)
  also from ∗ have eval-fds (fds-perzeta (1 / 2)) 1 = perzeta (1 / 2) 1
    by (simp add: perzeta-def)
  finally show (λn. −complex-of-real ((−1) ^ n / Suc n)) sums perzeta (1 / 2) 1
    by simp
  hence (λn. −complex-of-real ((−1) ^ n / Suc n)) sums −of-real (ln 2)
    by (intro sums-minus sums-of-real alternating-harmonic-series-sums)
  thus (λn. −complex-of-real ((−1) ^ n / Suc n)) sums −(ln 2)
    by (simp flip: Ln-of-real)
qed
```

2.9 Hurwitz's formula

We now move on to prove Hurwitz's formula relating the Hurwitz zeta function and the periodic zeta function. We mostly follow Apostol's proof, although we do make some small changes in order to make the proof more amenable to Isabelle's complex analysis library.

The big difference is that Apostol integrates along a circle with a slit, where the two sides of the slit lie on different branches of the integrand. This makes sense when looking as the integrand as a Riemann surface, but we do not have a notion of Riemann surfaces in Isabelle.

It is therefore much easier to simply cut the circle into an upper and a lower half. In fact, the integral on the lower half can be reduced to the one on the upper half easily by symmetry, so we really only need to handle the integral on the upper half. The integration contour that we will use is therefore a semi-annulus in the upper half of the complex plane, centred around the origin.

Now, first of all, we prove the existence of an important improper integral that we will need later.

lemma *set-integrable-bigo*:
  fixes $f\ g :: real \Rightarrow 'a :: \{banach,\ real\text{-}normed\text{-}field,\ second\text{-}countable\text{-}topology\}$
  assumes $f \in O(\lambda x.\ g\ x)$ **and** *set-integrable lborel* $\{a..\}$ g
  assumes $\bigwedge b.\ b \geq a \Longrightarrow$ *set-integrable lborel* $\{a..<b\}$ f
  assumes [*measurable*]: *set-borel-measurable borel* $\{a..\}$ f
  shows *set-integrable lborel* $\{a..\}$ f
proof −
  from *assms*(*1*) **obtain** $C\ x0$ **where** C: $C > 0\ \bigwedge x.\ x \geq x0 \Longrightarrow norm\ (f\ x) \leq C * norm\ (g\ x)$
    by (*fastforce elim*!: *landau-o.bigE simp*: *eventually-at-top-linorder*)
  define $x0'$ **where** $x0' = max\ a\ x0$

  have *set-integrable lborel* $\{a..<x0'\}$ f
    by (*intro assms*) (*auto simp*: $x0'$*-def*)
  moreover have *set-integrable lborel* $\{x0'..\}$ f **unfolding** *set-integrable-def*
  proof (*rule Bochner-Integration.integrable-bound*)
    from *assms*(*2*) **have** *set-integrable lborel* $\{x0'..\}$ g
      by (*rule set-integrable-subset*) (*auto simp*: $x0'$*-def*)
      thus *integrable lborel* $(\lambda x.\ C *_R (indicator\ \{x0'..\}\ x *_R g\ x))$ **unfolding** *set-integrable-def*
      by (*intro integrable-scaleR-right*) (*simp add*: *abs-mult norm-mult*)
  next
    from *assms*(*4*) **have** *set-borel-measurable borel* $\{x0'..\}$ f
      by (*rule set-borel-measurable-subset*) (*auto simp*: $x0'$*-def*)
    thus $(\lambda x.\ indicator\ \{x0'..\}\ x *_R f\ x) \in borel\text{-}measurable\ lborel$
      by (*simp add*: *set-borel-measurable-def*)
  next
    show $AE\ x\ in\ lborel.\ norm\ (indicator\ \{x0'..\}\ x *_R f\ x)$

$\leq$ *norm* ($C *_R$ (*indicator* {*x0'*..} *x* $*_R$ *g x*))
using *C* **by** (*intro AE-I2*) (*auto simp*: *abs-mult indicator-def x0'-def*)
qed
ultimately have *set-integrable lborel* ({*a*..<*x0'*} $\cup$ {*x0'*..}) *f*
by (*rule set-integrable-Un*) *auto*
also have {*a*..<*x0'*} $\cup$ {*x0'*..} = {*a*..} **by** (*auto simp*: *x0'-def*)
finally show *?thesis* **.**
qed

lemma *set-integrable-Gamma-hurwitz-aux2-real*:
fixes *s a* :: *real*
assumes *r* > *0* **and** *a* > *0*
shows *set-integrable lborel* {*r*..} (λ*x*. *x powr s* $*$ (*exp* ($-a * x$)) / (*1* $-$ *exp* ($-x$)))
(**is** *set-integrable - - ?g*)
proof (*rule set-integrable-bigo*)
have (λ*x*. *exp* ($-(a/2) * x$)) *integrable-on* {*r*..} **using** *assms*
by (*intro integrable-on-exp-minus-to-infinity*) *auto*
hence *set-integrable lebesgue* {*r*..} (λ*x*. *exp* ($-(a/2) * x$))
by (*intro nonnegative-absolutely-integrable*) *simp-all*
thus *set-integrable lborel* {*r*..} (λ*x*. *exp* ($-(a/2) * x$))
by (*simp add*: *set-integrable-def integrable-completion*)
next
fix *y* :: *real*
have *set-integrable lborel* {*r*..*y*} *?g* **using** *assms*
by (*intro borel-integrable-atLeastAtMost'*) (*auto intro*!: *continuous-intros*)
thus *set-integrable lborel* {*r*..<*y*} *?g*
by (*rule set-integrable-subset*) *auto*
next
from *assms* **show** *?g* $\in O(\lambda x.\ exp\ (-(a/2) * x))$
by *real-asymp*
qed (*auto simp*: *set-borel-measurable-def*)

lemma *set-integrable-Gamma-hurwitz-aux2*:
fixes *s* :: *complex* **and** *a* :: *real*
assumes *r* > *0* **and** *a* > *0*
shows *set-integrable lborel* {*r*..} (λ*x*. *x powr* $-s *$ (*exp* ($-$ *a* $*$ *x*)) / (*1* $-$ *exp* ($-$ *x*)))
(**is** *set-integrable - - ?g*)
proof $-$
have *set-integrable lborel* {*r*..} (λ*x*. *x powr* $-$*Re s* $*$ (*exp* ($-a * x$)) / (*1* $-$ *exp* ($-x$)))
(**is** *set-integrable - - ?g'*)
by (*rule set-integrable-Gamma-hurwitz-aux2-real*) (*use assms* **in** *auto*)
also have *?this* $\longleftrightarrow$ *integrable lborel* (λ*x*. *indicator* {*r*..} *x* $*_R$ *?g' x*)
by (*simp add*: *set-integrable-def*)
also have (λ*x*. *indicator* {*r*..} *x* $*_R$ *?g' x*) = (λ*x*. *norm* (*indicator* {*r*..} *x* $*_R$ *?g x*)) **using** *assms*
by (*auto simp*: *indicator-def norm-mult norm-divide norm-powr-real-powr fun-eq-iff exp-of-real exp-minus norm-inverse in-Reals-norm power2-eq-square*

divide-simps)
  finally show *?thesis* **unfolding** *set-integrable-def*
    by (*subst* (*asm*) *integrable-norm-iff*) *auto*
qed

lemma *closed-neg-Im-slit*: *closed* {*z*. *Re z* = *0* ∧ *Im z* ≤ *0*}
proof −
  have *closed* ({*z*. *Re z* = *0*} ∩ {*z*. *Im z* ≤ *0*})
    by (*intro closed-Int closed-halfspace-Re-eq closed-halfspace-Im-le*)
  also have {*z*. *Re z* = *0*} ∩ {*z*. *Im z* ≤ *0*} = {*z*. *Re z* = *0* ∧ *Im z* ≤ *0*} **by** *blast*
  finally show *?thesis* **.**
qed

We define tour semi-annulus path. When this path is mirrored into the lower half of the complex plane and subtracted from the original path and the outer radius tends to ∞, this becomes a Hankel contour extending to $-\infty$.

definition *hankel-semiannulus* :: *real* ⇒ *nat* ⇒ *real* ⇒ *complex* **where**
 hankel-semiannulus r N = (*let R* = (*2* ∗ *N* + *1*) ∗ *pi in*
    part-circlepath 0 R 0 pi +++ — Big half circle
    linepath (*of-real* (−*R*)) (*of-real* (−*r*)) +++ — Line on the negative real axis
    part-circlepath 0 r pi 0 +++ — Small half circle
    linepath (*of-real r*) (*of-real R*)) — Line on the positive real axis

lemma *path-hankel-semiannulus* [*simp*, *intro*]: *path* (*hankel-semiannulus r R*)
  and *valid-path-hankel-semiannulus* [*simp*, *intro*]: *valid-path* (*hankel-semiannulus r R*)
  and *pathfinish-hankel-semiannulus* [*simp*, *intro*]:
      pathfinish (*hankel-semiannulus r R*) = *pathstart* (*hankel-semiannulus r R*)
  by (*simp-all add*: *hankel-semiannulus-def Let-def*)

We set the stage for an application of the Residue Theorem. We define a function

$$f(s,z) = z^{-s}\frac{\exp(az)}{1-\exp(-z)}\ ,$$

which will be the integrand. However, the principal branch of z^{-s} has a branch cut along the non-positive real axis, which is bad because a part of our integration path also lies on the non-positive real axis. We therefore choose a slightly different branch of z^{-s} by moving the logarithm branch along by 90° so that the branch cut lies on the non-positive imaginary axis instead.

context
  fixes *a* :: *real*
  fixes *f* :: *complex* ⇒ *complex* ⇒ *complex*
    and *g* :: *complex* ⇒ *real* ⇒ *complex*
    and *h* :: *real* ⇒ *complex* ⇒ *real* ⇒ *complex*
    and *Res* :: *complex* ⇒ *nat* ⇒ *complex*

and Ln' :: *complex* ⇒ *complex*
and F :: *real* ⇒ *complex* ⇒ *complex*
assumes *a*: $a \in \{0{<}..1\}$

— Our custom branch of the logarithm
defines $Ln' \equiv (\lambda z.\ ln\ (-\mathrm{i} * z) + \mathrm{i} * pi\ /\ 2)$

— The integrand
defines $f \equiv (\lambda s\ z.\ exp\ (Ln'\ z * (-s) + of\text{-}real\ a * z)\ /\ (1 - exp\ z))$

— The integrand on the negative real axis
defines $g \equiv (\lambda s\ x.\ complex\text{-}of\text{-}real\ x\ powr\ {-s} * of\text{-}real\ (exp\ (-a{*}x))\ /\ of\text{-}real\ (1 - exp\ (-x)))$

— The integrand on the circular arcs
defines $h \equiv (\lambda r\ s\ t.\ r * \mathrm{i} * cis\ t * exp\ (a * (r * cis\ t) - (ln\ r + \mathrm{i} * t) * s)\ /$
$(1 - exp\ (r * cis\ t)))$

— The interesting part of the residues
defines $Res \equiv (\lambda s\ k.\ exp\ (of\text{-}real\ (2 * real\ k * pi * a) * \mathrm{i}) *$
$of\text{-}real\ (2 * real\ k * pi)\ powr\ (-s))$

— The periodic zeta function (at least on $\mathfrak{R}(s) > 1$ half-plane)
defines $F \equiv (\lambda q.\ eval\text{-}fds\ (fds\text{-}perzeta\ q))$
begin

First, some basic properties of our custom branch of the logarithm:

lemma *Ln'-i*: $Ln'\ \mathrm{i} = \mathrm{i} * pi\ /\ 2$
by (*simp add*: *Ln'-def*)

lemma *Ln'-of-real-pos*:
assumes $x > 0$
shows $Ln'\ (of\text{-}real\ x) = of\text{-}real\ (ln\ x)$
proof −
have $Ln'\ (of\text{-}real\ x) = Ln\ (of\text{-}real\ x * (-\mathrm{i})) + \mathrm{i} * pi\ /\ 2$
by (*simp add*: *Ln'-def mult-ac*)
also have $\ldots = of\text{-}real\ (ln\ x)$ **using** *assms*
by (*subst Ln-times-of-real*) (*auto simp*: *Ln-of-real*)
finally show *?thesis* **.**
qed

lemma *Ln'-of-real-neg*:
assumes $x < 0$
shows $Ln'\ (of\text{-}real\ x) = of\text{-}real\ (ln\ (-x)) + \mathrm{i} * pi$
proof −
have $Ln'\ (of\text{-}real\ x) = Ln\ (of\text{-}real\ (-x) * \mathrm{i}) + \mathrm{i} * pi\ /\ 2$
by (*simp add*: *Ln'-def mult-ac*)
also have $\ldots = of\text{-}real\ (ln\ (-x)) + \mathrm{i} * pi$ **using** *assms*
by (*subst Ln-times-of-real*) (*auto simp*: *Ln-Reals-eq*)

finally show *?thesis* **.**
qed

lemma *Ln′-times-of-real*:
Ln′ (*of-real x* ∗ *z*) = *of-real* (*ln x*) + *Ln′ z* **if** *x* > *0 z* ≠ *0* **for** *z x*
proof −
have *Ln′* (*of-real x* ∗ *z*) = *Ln* (*of-real x* ∗ (− i ∗ *z*)) + i ∗ *pi* / *2*
by (*simp add*: *Ln′-def mult-ac*)
also have ... = *of-real* (*ln x*) + *Ln′ z*
using *that* **by** (*subst Ln-times-of-real*) (*auto simp*: *Ln′-def Ln-of-real*)
finally show *?thesis* **.**
qed

lemma *Ln′-cis*:
assumes *t* ∈ {−*pi* / *2*<..*3* / *2* ∗ *pi*}
shows *Ln′* (*cis t*) = i ∗ *t*
proof −
have *exp* (i ∗ *pi* / *2*) = i **by** (*simp add*: *exp-eq-polar*)
hence *Ln* (− (i ∗ *cis t*)) = i ∗ (*t* − *pi* / *2*) **using** *assms*
by (*intro Ln-unique*) (*auto simp*: *algebra-simps exp-diff cis-conv-exp*)
thus *?thesis* **by** (*simp add*: *Ln′-def algebra-simps*)
qed

Next, we show that the line and circle integrals are holomorphic using Leibniz's rule:

lemma *contour-integral-part-circlepath-h*:
assumes *r*: *r* > *0*
shows *contour-integral* (*part-circlepath 0 r 0 pi*) (*f s*) = *integral* {*0*..*pi*} (*h r s*)
proof −
have *contour-integral* (*part-circlepath 0 r 0 pi*) (*f s*) =
integral {*0*..*pi*} (λ*t*. *f s* (*r* ∗ *cis t*) ∗ *r* ∗ i ∗ *cis t*)
by (*simp add*: *contour-integral-part-circlepath-eq*)
also have *integral* {*0*..*pi*} (λ*t*. *f s* (*r* ∗ *cis t*) ∗ *r* ∗ i ∗ *cis t*) = *integral* {*0*..*pi*} (*h r s*)
proof (*intro integral-cong*)
fix *t* **assume** *t*: *t* ∈ {*0*..*pi*}
have −(*pi* / *2*) < *0* **by** *simp*
also have *0* ≤ *t* **using** *t* **by** *simp*
finally have *t* ∈ {−(*pi*/*2*)<..*3*/*2*∗*pi*} **using** *t* **by** *auto*
thus *f s* (*r* ∗ *cis t*) ∗ *r* ∗ i ∗ *cis t* = *h r s t*
using *r* **by** (*simp add*: *f-def Ln′-times-of-real Ln′-cis h-def Ln-Reals-eq*)
qed
finally show *?thesis* **.**
qed

lemma *integral-g-holomorphic*:
assumes *b* > *0*
shows (λ*s*. *integral* {*b*..*c*} (*g s*)) *holomorphic-on A*
proof −

define g' **where** $g' = (\lambda s\ t.\ -\ (\textit{of-real}\ t\ \textit{powr}\ (-s)) * \textit{complex-of-real}\ (\textit{ln}\ t) *$
$\textit{of-real}\ (\textit{exp}\ (-\ (a * t)))\ /\ \textit{of-real}\ (\mathit{1}\ -\ \textit{exp}\ (-\ t)))$
have $(\lambda s.\ \textit{integral}\ (\textit{cbox}\ b\ c)\ (g\ s))$ *holomorphic-on UNIV*
proof (*rule leibniz-rule-holomorphic*)
fix s :: *complex* **and** t :: *real* **assume** $t \in \textit{cbox}\ b\ c$
thus $((\lambda s.\ g\ s\ t)$ *has-field-derivative* $g'\ s\ t)$ (*at* s) **using** *assms*
by (*auto simp*: *g-def* g'*-def powr-def Ln-Reals-eq intro*!: *derivative-eq-intros*)
next
fix s **show** $g\ s$ *integrable-on cbox* $b\ c$ **for** s **unfolding** *g-def* **using** *assms*
by (*intro integrable-continuous continuous-intros*) *auto*
next
show *continuous-on* (*UNIV* $\times$ *cbox* $b\ c$) $(\lambda(s,\ t).\ g'\ s\ t)$ **using** *assms*
by (*auto simp*: g'*-def case-prod-unfold intro*!: *continuous-intros*)
qed *auto*
thus *?thesis* **by** *auto*
qed

lemma *integral-h-holomorphic*:
assumes r: $r \in \{\mathit{0}{<}..{<}\mathit{2}\}$
shows $(\lambda s.\ \textit{integral}\ \{b..c\}\ (h\ r\ s))$ *holomorphic-on* A
proof $-$
have *no-sing*: $\textit{exp}\ (r * \textit{cis}\ t) \neq \mathit{1}$ **for** t
proof
define z **where** $z = r * \textit{cis}\ t$
assume $\textit{exp}\ z = \mathit{1}$
then obtain n **where** $\textit{norm}\ z = \mathit{2} * \textit{pi} * \textit{of-int}\ |n|$
by (*auto simp*: *exp-eq-1 cmod-def abs-mult*)
moreover have $\textit{norm}\ z = r$ **using** r **by** (*simp add*: *z-def norm-mult*)
ultimately have *r-eq*: $r = \mathit{2} * \textit{pi} * \textit{of-int}\ |n|$ **by** *simp*
with r **have** $n \neq \mathit{0}$ **by** *auto*
moreover from r **have** $r < \mathit{2} * \textit{pi}$ **using** *pi-gt3* **by** *simp*
with *r-eq* **have** $|n| < \mathit{1}$ **by** *simp*
ultimately show *False* **by** *simp*
qed

define h' **where** $h' = (\lambda s\ t.\ \textit{exp}\ (a * r * \textit{cis}\ t - (\textit{ln}\ r + \mathrm{i} * t) * s) *$
$(-\textit{ln}\ r - \mathrm{i} * t) * (r * \mathrm{i} * \textit{cis}\ t)\ /\ (\mathit{1} - \textit{exp}\ (r * \textit{cis}\ t)))$
have $(\lambda s.\ \textit{integral}\ (\textit{cbox}\ b\ c)\ (h\ r\ s))$ *holomorphic-on UNIV*
proof (*rule leibniz-rule-holomorphic*)
fix $s\ t$ **assume** $t \in \textit{cbox}\ b\ c$
thus $((\lambda s.\ h\ r\ s\ t)$ *has-field-derivative* $h'\ s\ t)$ (*at* s) **using** *no-sing* r
by (*auto intro*!: *derivative-eq-intros simp*: *h-def* h'*-def mult-ac Ln-Reals-eq*)
next
fix s **show** $h\ r\ s$ *integrable-on cbox* $b\ c$ **using** *no-sing* **unfolding** *h-def*
by (*auto intro*!: *integrable-continuous-real continuous-intros*)
next
show *continuous-on* (*UNIV* $\times$ *cbox* $b\ c$) $(\lambda(s,\ t).\ h'\ s\ t)$ **using** *no-sing*
by (*auto simp*: h'*-def case-prod-unfold intro*!: *continuous-intros*)
qed *auto*

thus *?thesis* **by** *auto*
qed

We now move on to the core result, which uses the Residue Theorem to relate a contour integral along a semi-annulus to a partial sum of the periodic zeta function.

lemma *hurwitz-formula-integral-semiannulus*:
fixes N :: *nat* **and** r :: *real* **and** s :: *complex*
defines $R \equiv$ *real* $(2 * N + 1) * pi$
assumes $r > 0$ **and** $r < 2$
shows *exp* $(-\mathrm{i} * pi * s) *$ *integral* $\{r..R\}$ $(\lambda x.\ x$ *powr* $(-s) *$ *exp* $(-a * x)$ / $(1 -$ *exp* $(-x))) +$
integral $\{r..R\}$ $(\lambda x.\ x$ *powr* $(-s) *$ *exp* $(a * x)$ / $(1 -$ *exp* $x)) +$
contour-integral (*part-circlepath* 0 R 0 *pi*) (f s) +
contour-integral (*part-circlepath* 0 r *pi* 0) (f s)
$= -2 * pi * \mathrm{i} *$ *exp* $(- s *$ *of-real pi* $* \mathrm{i} / 2) * (\sum k \in \{0<..N\}.$ *Res* s $k)$
(**is** *?thesis1*)
and f s *contour-integrable-on hankel-semiannulus* r N
proof $-$
have $2 < 1 * pi$ **using** *pi-gt3* **by** *simp*
also have $\ldots \leq R$ **unfolding** *R-def* **by** (*intro mult-right-mono*) *auto*
finally have $R > 2$ **by** (*auto simp*: *R-def*)
note *r-R* = ‹$r > 0$› ‹$r < 2$› *this*

— We integrate along the edge of a semi-annulus in the upper half of the complex plane. It consists of a big semicircle, a small semicircle, and two lines connecting the two circles, one on the positive real axis and one on the negative real axis. The integral along the big circle will vanish as the radius of the circle tends to ∞, whereas the remaining path becomes a Hankel contour, and the integral along that Hankel contour is what we are interested in, since it is connected to the Hurwitz zeta function.
define *big-circle* **where** *big-circle* = *part-circlepath* 0 R 0 *pi*
define *small-circle* **where** *small-circle* = *part-circlepath* 0 r *pi* 0
define *neg-line* **where** *neg-line* = *linepath* (*complex-of-real* $(-R)$) (*complex-of-real* $(-r)$)
define *pos-line* **where** *pos-line* = *linepath* (*complex-of-real* r) (*complex-of-real* R)
define γ **where** γ = *hankel-semiannulus* r N
have γ*-altdef*: γ = *big-circle* +++ *neg-line* +++ *small-circle* +++ *pos-line*
by (*simp add*: γ*-def R-def add-ac hankel-semiannulus-def big-circle-def neg-line-def small-circle-def pos-line-def*)
have [*simp*]: *path* γ *valid-path* γ *pathfinish* γ = *pathstart* γ
by (*simp-all add*: γ*-def*)

— The integrand has a branch cut along the non-positive imaginary axis and additional simple poles at $2n\pi i$ for any $n \in \mathbb{N}_{>0}$. The radius of the smaller circle will always be less than 2π and the radius of the bigger circle of the form $(2N+1)\pi$, so we always have precisely the first N poles inside our path.
define *sngs* **where** *sngs* = $(\lambda n.$ *of-real* $(2 * pi *$ *real* $n) * \mathrm{i})$ ‘ $\{0<..\}$

define *sngs'* **where** *sngs'* = (λ*n. of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i) ‘ {*0*<..*N*}
have *sngs-subset*: *sngs'* ⊆ *sngs* **unfolding** *sngs-def sngs'-def* **by** (*intro image-mono*) *auto*
have *closed-sngs* [*intro*]: *closed* (*sngs* − *sngs'*) **unfolding** *sngs-def*
proof (*rule discrete-imp-closed*[*of 2* ∗ *pi*]; *safe?*)
fix *m n* :: *nat*
assume *dist* (*of-real* (*2* ∗ *pi* ∗ *real m*) ∗ i) (*of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i) < *2* ∗ *pi*
also have *dist* (*of-real* (*2* ∗ *pi* ∗ *real m*) ∗ i) (*of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i) =
norm (*of-real* (*2* ∗ *pi* ∗ *real m*) ∗ i − *of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i)
by (*simp add*: *dist-norm*)
also have *of-real* (*2* ∗ *pi* ∗ *real m*) ∗ i − *of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i =
of-real (*2* ∗ *pi*) ∗ i ∗ *of-int* (*int m* − *int n*) **by** (*simp add*: *algebra-simps*)
also have *norm* ... = *2* ∗ *pi* ∗ *of-int* |*int m* − *int n*|
unfolding *norm-mult norm-of-int* **by** (*simp add*: *norm-mult*)
finally have |*real m* − *real n*| < *1* **by** *simp*
hence *m* = *n* **by** *linarith*
thus *of-real* (*2* ∗ *pi* ∗ *real m*) ∗ i = *of-real* (*2* ∗ *pi* ∗ *real n*) ∗ i **by** *simp*
qed *auto*

— We define an area within which the integrand is holomorphic. Choosing this area as big as possible makes things easier later on, so we only remove the branch cut and the poles.
define *S* **where** *S* = − {*z*. *Re z* = *0* ∧ *Im z* ≤ *0*} − (*sngs* − *sngs'*)
define *S'* **where** *S'* = − {*z*. *Re z* = *0* ∧ *Im z* ≤ *0*}

have *sngs*: *exp z* = *1* ⟷ *z* ∈ *sngs* **if** *Re z* ≠ *0* ∨ *Im z* > *0* **for** *z*
proof
assume *exp z* = *1*
then obtain *n* **where** *n*: *z* = *2* ∗ *pi* ∗ *of-int n* ∗ i
unfolding *exp-eq-1* **by** (*auto simp*: *complex-eq-iff mult-ac*)
moreover from *n* **and** *pi-gt-zero* **and** *that* **have** *n* > *0* **by** (*auto simp*: *zero-less-mult-iff*)
ultimately have *z* = *of-real* (*2* ∗ *pi* ∗ *nat n*) ∗ i **and** *nat n* ∈ {*0*<..}
by *auto*
thus *z* ∈ *sngs* **unfolding** *sngs-def* **by** *blast*
qed (*insert that*, *auto simp*: *sngs-def exp-eq-polar*)

— We show that the path stays within the well-behaved area.
have *path-image neg-line* = *of-real* ‘ {−*R*..−*r*} **using** *r-R*
by (*auto simp*: *neg-line-def closed-segment-Reals closed-segment-eq-real-ivl*)
hence *path-image neg-line* ⊆ *S* − *sngs'* **using** *r-R sngs-subset*
by (*auto simp*: *S-def sngs-def complex-eq-iff*)

have *path-image pos-line* = *of-real* ‘ {*r*..*R*} **using** *r-R*
by (*auto simp*: *pos-line-def closed-segment-Reals closed-segment-eq-real-ivl*)
hence *path-image pos-line* ⊆ *S* − *sngs'* **using** *r-R sngs-subset*
by (*auto simp*: *S-def sngs-def complex-eq-iff*)

have *part-circlepath-in-S*: $z \in S - sngs'$
if $z \in$ *path-image* (*part-circlepath* $0\ r'\ 0\ pi$) $\vee$ $z \in$ *path-image* (*part-circlepath* $0\ r'\ pi\ 0$)
and $r' > 0$ $r' \notin (\lambda n.\ 2 * pi * real\ n)$ ' $\{0<..\}$ **for** $z\ r'$
proof $-$
have z: $norm\ z = r' \wedge Im\ z \geq 0$ **using** *that*
by (*auto simp*: *path-image-def part-circlepath-def norm-mult Im-exp linepath-def intro*!: *mult-nonneg-nonneg sin-ge-zero*)
hence $Re\ z \neq 0 \vee Im\ z > 0$ **using** *that* **by** (*auto simp*: *cmod-def*)
moreover from z **and** *that* **have** $z \notin sngs$
by (*auto simp*: *sngs-def norm-mult image-iff*)
ultimately show $z \in S - sngs'$ **using** *sngs-subset* **by** (*auto simp*: *S-def*)
qed

{
fix n :: *nat* **assume** n: $n > 0$
have $r < 2 * pi * 1$ **using** *pi-gt3 r-R* **by** *simp*
also have $\ldots \leq 2 * pi * real\ n$ **using** n **by** (*intro mult-left-mono*) *auto*
finally have $r < \ldots$ **.**
}
hence $r \notin (\lambda n.\ 2 * pi * real\ n)$ ' $\{0<..\}$ **using** *r-R* **by** *auto*
from *part-circlepath-in-S*[*OF - - this*] *r-R* **have** *path-image small-circle* $\subseteq S - sngs'$
by (*auto simp*: *small-circle-def*)

{
fix n :: *nat* **assume** n: $n > 0$ $2 * pi * real\ n = real\ (2 * N + 1) * pi$
hence $real\ (2 * n) = real\ (2 * N + 1)$ **unfolding** *of-nat-mult* **by** *simp*
hence *False* **unfolding** *of-nat-eq-iff* **by** *presburger*
}
hence $R \notin (\lambda n.\ 2 * pi * real\ n)$ ' $\{0<..\}$ **unfolding** *R-def* **by** *force*
from *part-circlepath-in-S*[*OF - - this*] *r-R* **have** *path-image big-circle* $\subseteq S - sngs'$
by (*auto simp*: *big-circle-def*)

note *path-images* =
‹*path-image small-circle* $\subseteq S - sngs'$› ‹*path-image big-circle* $\subseteq S - sngs'$›
‹*path-image neg-line* $\subseteq S - sngs'$› ‹*path-image pos-line* $\subseteq S - sngs'$›
have *path-image* $\gamma \subseteq S - sngs'$ **using** *path-images*
by (*auto simp*: γ*-altdef path-image-join big-circle-def neg-line-def small-circle-def pos-line-def*)

— We need to show that the complex plane is still connected after we removed the branch cut and the poles. We do this by showing that the complex plane with the branch cut removed is starlike and therefore connected. Then we remove the (countably many) poles, which does not break connectedness either.
have *open* S **using** *closed-neg-Im-slit* **by** (*auto simp*: *S-def*)
have *starlike* ($UNIV - \{z.\ Re\ z = 0 \wedge Im\ z \leq 0\}$)
(**is** *starlike* $?S'$) **unfolding** *starlike-def*
proof (*rule bexI ballI*)+

have *1* $\leq$ *1* $*$ *pi* **using** *pi-gt3* **by** *simp*
also have ... $<$ (*2* + *2* $*$ *real N*) $*$ *pi* **by** (*intro mult-strict-right-mono*) *auto*
finally show $*$: i $\in$ *?S'* **by** (*auto simp*: *S-def*)
fix *z* **assume** *z*: *z* $\in$ *?S'*
have *closed-segment* i *z* $\cap$ {*z*. *Re z* = *0* $\wedge$ *Im z* $\leq$ *0*} = {}
proof *safe*
fix *s* **assume** *s*: *s* $\in$ *closed-segment* i *z Re s* = *0 Im s* $\leq$ *0*
then obtain *t* **where** *t*: *t* $\in$ {*0..1*} *s* = *linepath* i *z t*
using *linepath-image-01* **by** *blast*
with *z s t* **have** *z'*: *Re z* = *0 Im z* $>$ *0*
by (*auto simp*: *Re-linepath' S-def linepath-0'*)
with *s* **have** *Im s* $\in$ *closed-segment 1* (*Im z*) $\wedge$ *Im s* $\leq$ *0*
by (*subst* (*asm*) *closed-segment-same-Re*) *auto*
with *z'* **show** *s* $\in$ {}
by (*auto simp*: *closed-segment-eq-real-ivl split*: *if-splits*)
qed
thus *closed-segment* i *z* $\subseteq$ *?S'* **by** (*auto simp*: *S-def*)
qed
hence *connected ?S'* **by** (*rule starlike-imp-connected*)
hence *connected S'* **by** (*simp add*: *Compl-eq-Diff-UNIV S'-def*)
have *connected S* **unfolding** *S-def*
by (*rule connected-open-diff-countable*)
(*insert* ‹*connected S'*›, *auto simp*: *sngs-def closed-neg-Im-slit S'-def*)

— The integrand is now clearly holomorphic on *S* − *sngs'* and we can apply the Residue Theorem.
have *holo*: *f s holomorphic-on* (*S* − *sngs'*)
unfolding *f-def Ln'-def S-def* **using** *sngs*
by (*auto intro*!: *holomorphic-intros simp*: *complex-nonpos-Reals-iff*)
have *contour-integral* γ (*f s*) =
of-real (*2* $*$ *pi*) $*$ i $*$ ($\sum$ *z*$\in$*sngs'*. *winding-number* γ *z* $*$ *residue* (*f s*) *z*)
proof (*rule Residue-theorem*)
show $\forall$ *z*. *z* $\notin$ *S* $\longrightarrow$ *winding-number* γ *z* = *0*
proof *safe*
fix *z* **assume** *z* $\notin$ *S*
hence *Re z* = *0* $\wedge$ *Im z* $\leq$ *0* $\vee$ *z* $\in$ *sngs* − *sngs'* **by** (*auto simp*: *S-def*)
thus *winding-number* γ *z* = *0*
proof
define *x* **where** *x* = −*Im z*
assume *Re z* = *0* $\wedge$ *Im z* $\leq$ *0*
hence *x*: *z* = −*of-real x* $*$ i *x* $\geq$ *0* **unfolding** *complex-eq-iff* **by** (*simp-all add*: *x-def*)
obtain *B* **where** $\bigwedge$*z*. *norm z* $\geq$ *B* $\Longrightarrow$ *winding-number* γ *z* = *0*
using *winding-number-zero-at-infinity*[*of* γ] **by** *auto*
hence *winding-number* γ (−*of-real* (*max B 0*) $*$ i) = *0* **by** (*auto simp*: *norm-mult*)
also have *winding-number* γ (−*of-real* (*max B 0*) $*$ i) = *winding-number* γ *z*
proof (*rule winding-number-eq*)

from x **have** *closed-segment* $(-$*of-real* $($*max B 0*$) * \mathrm{i})$ $z \subseteq \{z.$ *Re* $z = 0$ $\wedge$ *Im* $z \leq 0\}$
by (*auto simp*: *closed-segment-same-Re closed-segment-eq-real-ivl*)
with ‹*path-image* $\gamma \subseteq S - sngs'$›
show *closed-segment* $(-$*of-real* $($*max B 0*$) * \mathrm{i})$ $z \cap$ *path-image* $\gamma = \{\}$
by (*auto simp*: *S-def*)
qed *auto*
finally show *winding-number* γ $z = 0$ **.**
next
assume z: $z \in sngs - sngs'$
show *winding-number* γ $z = 0$
proof (*rule winding-number-zero-outside*)
have *path-image* γ = *path-image big-circle* $\cup$ *path-image neg-line* $\cup$
path-image small-circle $\cup$ *path-image pos-line*
unfolding γ*-altdef small-circle-def big-circle-def pos-line-def neg-line-def*
by (*simp add*: *path-image-join Un-assoc*)
also have ... $\subseteq$ *cball 0* $((2 * N + 1) * pi)$ **using** *r-R*
by (*auto simp*: *small-circle-def big-circle-def pos-line-def neg-line-def*
path-image-join norm-mult R-def path-image-part-circlepath'
in-Reals-norm closed-segment-Reals closed-segment-eq-real-ivl)
finally show *path-image* $\gamma \subseteq$ ... **.**
qed (*insert z*, *auto simp*: *sngs-def sngs'-def norm-mult*)
qed
qed
qed (*insert* ‹*path-image* $\gamma \subseteq S - sngs'$› ‹*connected S*› ‹*open S*› *holo*, *auto simp*: *sngs'-def*)

— We can use Wenda Li's framework to compute the winding numbers at the poles and show that they are all 1.
also have *winding-number* γ $z = 1$ **if** $z \in sngs'$ **for** z
proof −
have $r < 2 * pi * 1$ **using** *pi-gt3 r-R* **by** *simp*
also have ... $\leq 2 * pi *$ *real* n **if** $n > 0$ **for** n **using** *that* **by** (*intro mult-left-mono*) *auto*
finally have *norm-z*: *norm* $z > r$ *norm* $z < R$ **using** *that r-R*
by (*auto simp*: *sngs'-def norm-mult R-def*)

have *cindex-pathE big-circle* $z = -1$ **using** *r-R that* **unfolding** *big-circle-def*
by (*subst cindex-pathE-circlepath-upper*(*1*)) (*auto simp*: *sngs'-def norm-mult R-def*)
have *cindex-pathE small-circle* $z = -1$ **using** *r-R that norm-z* **unfolding** *small-circle-def*
by (*subst cindex-pathE-reversepath'*, *subst reversepath-part-circlepath*,
subst cindex-pathE-circlepath-upper(*2*)) (*auto simp*: *sngs'-def norm-mult*)
have *cindex-pathE neg-line* $z = 0$ *cindex-pathE pos-line* $z = 0$
unfolding *neg-line-def pos-line-def* **using** *r-R that*
by (*subst cindex-pathE-linepath*; *force simp*: *neg-line-def cindex-pathE-linepath closed-segment-Reals closed-segment-eq-real-ivl sngs'-def complex-eq-iff*)+
note *indices* = ‹*cindex-pathE big-circle* $z = -1$› ‹*cindex-pathE small-circle* z

= −1›
‹cindex-pathE neg-line z = 0› ‹cindex-pathE pos-line z = 0›
show ?thesis **unfolding** γ-altdef big-circle-def small-circle-def pos-line-def neg-line-def
by eval-winding (insert indices path-images that,
auto simp: big-circle-def small-circle-def pos-line-def neg-line-def)
qed
hence (∑ z∈sngs′. winding-number γ z ∗ residue (f s) z) = (∑ z∈sngs′. residue (f s) z)
by simp
also have ... = (∑ k∈{0<..N}. residue (f s) (2 ∗ pi ∗ of-nat k ∗ i))
unfolding sngs′-def **by** (subst sum.reindex) (auto intro!: inj-onI simp: o-def)

— Next, we compute the residues at each pole.
also have residue (f s) (2 ∗ pi ∗ of-nat k ∗ i) = −exp (− s ∗ of-real pi ∗ i / 2) ∗ Res s k
if k ∈ {0<..N} **for** k **unfolding** f-def
proof (subst residue-simple-pole-deriv)
show open S′ **using** closed-neg-Im-slit **by** (auto simp: S′-def)
show connected S′ **by** fact
show (λz. exp (Ln′ z ∗ (−s) + of-real a ∗ z)) holomorphic-on S′
(λz. 1 − exp z) holomorphic-on S′
by (auto simp: S′-def Ln′-def complex-nonpos-Reals-iff intro!: holomorphic-intros)
have ((λz. 1 − exp z) has-field-derivative −exp (2 ∗ pi ∗ k ∗ i))
(at (of-real (2 ∗ pi ∗ real k) ∗ i))
by (auto intro!: derivative-eq-intros)
also have −exp (2 ∗ pi ∗ k ∗ i) = −1 **by** (simp add: exp-eq-polar)
finally show ((λz. 1 − exp z) has-field-derivative −1)
(at (of-real (2 ∗ pi ∗ real k) ∗ i)) **.**
have Im (of-real (2 ∗ pi ∗ real k) ∗ i) > 0 **using** pi-gt-zero that
by auto
thus of-real (2 ∗ pi ∗ real k) ∗ i ∈ S′ **by** (simp add: S′-def)

have exp (i ∗ pi / 2) = i **by** (simp add: exp-eq-polar)
hence exp (Ln′ (complex-of-real (2 ∗ pi ∗ real k) ∗ i) ∗ −s +
of-real a ∗ (of-real (2 ∗ pi ∗ real k) ∗ i)) / −1 =
− exp (2 ∗ k ∗ a ∗ pi ∗ i − s ∗ pi ∗ i / 2 − s ∗ ln (2 ∗ k ∗ pi)) (**is** ?R = -)
using that **by** (subst Ln′-times-of-real) (simp-all add: Ln′-i algebra-simps exp-diff)
also have ... = −exp (− s ∗ of-real pi ∗ i / 2) ∗ Res s k **using** that
by (simp add: Res-def exp-diff powr-def exp-minus inverse-eq-divide Ln-Reals-eq mult-ac)
finally show ?R = −exp (− s ∗ of-real pi ∗ i / 2) ∗ Res s k **.**
qed (insert that, auto simp: S′-def exp-eq-polar)
hence (∑ k∈{0<..N}. residue (f s) (2 ∗ pi ∗ of-nat k ∗ i)) =
−exp (− s ∗ of-real pi ∗ i / 2) ∗ (∑ k∈{0<..N}. Res s k)
by (simp add: sum-distrib-left)

— This gives us the final result:
finally have *contour-integral* γ (*f s*) =
$-2 * pi * \mathrm{i} * exp\ (- s * of\text{-}real\ pi * \mathrm{i}\ /\ 2) * (\sum k\in\{0<..N\}.\ Res\ s\ k)$ **by** *simp*

— Lastly, we decompose the contour integral into its four constituent integrals because this makes them somewhat nicer to work with later on.
also show *f s contour-integrable-on* γ
proof (*rule contour-integrable-holomorphic-simple*)
show *path-image* $\gamma \subseteq S - sngs'$ **by** *fact*
have *closed sngs′* **by** (*intro finite-imp-closed*) (*auto simp*: *sngs′-def*)
with ‹*open S*› **show** *open* (*S* − *sngs′*) **by** *auto*
qed (*insert holo*, *auto*)
hence *eq*: *contour-integral* γ (*f s*) =
contour-integral big-circle (*f s*) + *contour-integral neg-line* (*f s*) +
contour-integral small-circle (*f s*) + *contour-integral pos-line* (*f s*)
unfolding γ*-altdef big-circle-def neg-line-def small-circle-def pos-line-def* **by** *simp*

also have *contour-integral neg-line* (*f s*) = *integral* {−*R*..−*r*} (λ*x*. *f s* (*complex-of-real x*))
unfolding *neg-line-def* **using** *r-R* **by** (*subst contour-integral-linepath-Reals-eq*) *auto*
also have ... = $exp\ (- \mathrm{i} * pi * s) *$
$integral\ \{r..R\}\ (\lambda x.\ exp\ (-ln\ x * s) * exp\ (-a * x)\ /\ (1 - exp\ (-x)))$
(**is** - = - * *?I*) **unfolding** *integral-mult-right* [*symmetric*] **using** *r-R*
by (*subst Henstock-Kurzweil-Integration.integral-reflect-real* [*symmetric*], *intro integral-cong*)
(*auto simp*: *f-def exp-of-real Ln′-of-real-neg exp-minus exp-Reals-eq*
exp-diff exp-add field-simps)
also have *?I* = $integral\ \{r..R\}\ (\lambda x.\ x\ powr\ (-s) * exp\ (-a * x)\ /\ (1 - exp\ (-x)))$ **using** *r-R*
by (*intro integral-cong*) (*auto simp*: *powr-def Ln-Reals-eq exp-minus exp-diff field-simps*)

also have *contour-integral pos-line* (*f s*) = *integral* {*r*..*R*} (λ*x*. *f s* (*complex-of-real x*))
unfolding *pos-line-def* **using** *r-R* **by** (*subst contour-integral-linepath-Reals-eq*) *auto*
also have ... = $integral\ \{r..R\}\ (\lambda x.\ x\ powr\ (-s) * exp\ (a * x)\ /\ (1 - exp\ x))$
using *r-R* **by** (*intro integral-cong*) (*simp add*: *f-def Ln′-of-real-pos exp-diff exp-minus*
exp-Reals-eq field-simps powr-def Ln-Reals-eq)
finally show *?thesis1* **by** (*simp only*: *add-ac big-circle-def small-circle-def*)
qed

Next, we need bounds on the integrands of the two semicircles.

lemma *hurwitz-formula-bound1*:
  defines $H \equiv \lambda z.\ exp\ (complex\text{-}of\text{-}real\ a * z)\ /\ (1 - exp\ z)$
  assumes $r > 0$
  obtains C **where** $C \geq 0$ **and** $\bigwedge z.\ z \notin (\bigcup n{::}int.\ ball\ (2 * n * pi * \mathrm{i})\ r) \Longrightarrow norm\ (H\ z) \leq C$
proof −
  define A **where** $A = cbox\ (-1 - pi * \mathrm{i})\ (1 + pi * \mathrm{i}) - ball\ 0\ r$
  {
    fix z **assume** $z \in A$
    have $exp\ z \neq 1$
    proof
      assume $exp\ z = 1$
      then obtain $n :: int$ **where** [*simp*]: $z = 2 * n * pi * \mathrm{i}$
        by (*subst* (*asm*) *exp-eq-1*) (*auto simp*: *complex-eq-iff*)
      from ‹$z \in A$› **have** $(2 * n) * pi \geq (-1) * pi$ **and** $(2 * n) * pi \leq 1 * pi$
        by (*auto simp*: *A-def in-cbox-complex-iff*)
      hence $n = 0$ **by** (*subst* (*asm*) (*1 2*) *mult-le-cancel-right*) *auto*
      with ‹$z \in A$› **and** ‹$r > 0$› **show** *False* **by** (*simp add*: *A-def*)
    qed
  }
  hence *continuous-on A H*
    by (*auto simp*: *A-def H-def intro*!: *continuous-intros*)
  moreover have *compact A* **by** (*auto simp*: *A-def compact-eq-bounded-closed*)
  ultimately have *compact* (*H ‘ A*) **by** (*rule compact-continuous-image*)
  hence *bounded* (*H ‘ A*) **by** (*rule compact-imp-bounded*)
  then obtain C **where** *bound-inside*: $\bigwedge z.\ z \in A \Longrightarrow norm\ (H\ z) \leq C$
    by (*auto simp*: *bounded-iff*)

  have *bound-outside*: $norm\ (H\ z) \leq exp\ 1\ /\ (exp\ 1 - 1)$ **if** $|Re\ z| > 1$ **for** z
  proof −
    have $norm\ (H\ z) = exp\ (a * Re\ z)\ /\ norm\ (1 - exp\ z)$
      by (*simp add*: *H-def norm-divide*)
    also have $|1 - exp\ (Re\ z)| \leq norm\ (1 - exp\ z)$
      by (*rule order.trans*[*OF - norm-triangle-ineq3*]) *simp*
    hence $exp\ (a * Re\ z)\ /\ norm\ (1 - exp\ z) \leq exp\ (a * Re\ z)\ /\ |1 - exp\ (Re\ z)|$
      using *that* **by** (*intro divide-left-mono mult-pos-pos*) *auto*
    also have $\ldots \leq exp\ 1\ /\ (exp\ 1 - 1)$
    proof (*cases* $Re\ z > 1$)
      case *True*
      hence $exp\ (a * Re\ z)\ /\ |1 - exp\ (Re\ z)| = exp\ (a * Re\ z)\ /\ (exp\ (Re\ z) - 1)$
        by *simp*
      also have $\ldots \leq exp\ (Re\ z)\ /\ (exp\ (Re\ z) - 1)$
        using *a True* **by** (*intro divide-right-mono*) *auto*
      also have $\ldots = 1\ /\ (1 - exp\ (-Re\ z))$ **by** (*simp add*: *exp-minus field-simps*)
      also have $\ldots \leq 1\ /\ (1 - exp\ (-1))$ **using** *True* **by** (*intro divide-left-mono diff-mono*) *auto*
      also have $\ldots = exp\ 1\ /\ (exp\ 1 - 1)$ **by** (*simp add*: *exp-minus field-simps*)
      finally show *?thesis* **.**
    next

case *False*
with *that* **have** $Re\ z < -1$ **by** *simp*
hence $exp\ (a * Re\ z)\ /\ |1 - exp\ (Re\ z)| = exp\ (a * Re\ z)\ /\ (1 - exp\ (Re\ z))$ **by** *simp*
also have $\ldots \le 1\ /\ (1 - exp\ (Re\ z))$
using *a* **and** ‹$Re\ z < -1$› **by** (*intro divide-right-mono*) (*auto intro*: *mult-nonneg-nonpos*)
also have $\ldots \le 1\ /\ (1 - exp\ (-1))$
using ‹$Re\ z < -1$› **by** (*intro divide-left-mono*) *auto*
also have $\ldots = exp\ 1\ /\ (exp\ 1 - 1)$ **by** (*simp add*: *exp-minus field-simps*)
finally show *?thesis* **.**
qed
finally show *?thesis* **.**
qed

define *D* **where** $D = max\ C\ (exp\ 1\ /\ (exp\ 1 - 1))$
have $D \ge 0$ **by** (*simp add*: *D-def max.coboundedI2*)

have $norm\ (H\ z) \le D$ **if** $z \notin (\bigcup n{::}int.\ ball\ (2 * n * pi * \mathrm{i})\ r)$ **for** *z*
proof (*cases* $|Re\ z| \le 1$)
case *False*
with *bound-outside*[*of z*] **show** *?thesis* **by** (*simp add*: *D-def*)
next
case *True*
define *n* **where** $n = \lfloor Im\ z\ /\ (2 * pi) + 1\ /\ 2 \rfloor$

have $Im\ (z - 2 * n * pi * \mathrm{i}) = frac\ (Im\ z\ /\ (2 * pi) + 1\ /\ 2) * (2 * pi) - pi$
by (*simp add*: *n-def frac-def algebra-simps*)
also have $\ldots \in \{-pi..<pi\}$ **using** *frac-lt-1* **by** *simp*
finally have $norm\ (H\ (z - 2 * n * pi * \mathrm{i})) \le C$ **using** *True that*
by (*intro bound-inside*) (*auto simp*: *A-def in-cbox-complex-iff dist-norm n-def*)
also have $exp\ (2 * pi * n * \mathrm{i}) = 1$ **by** (*simp add*: *exp-eq-polar*)
hence $norm\ (H\ (z - 2 * n * pi * \mathrm{i})) = norm\ (H\ z)$
by (*simp add*: *H-def norm-divide exp-diff mult-ac*)
also have $C \le D$ **by** (*simp add*: *D-def*)
finally show *?thesis* **.**
qed
from ‹$D \ge 0$› **and** *this* **show** *?thesis* **by** (*rule that*)
qed

lemma *hurwitz-formula-bound2*:
obtains *C* **where** $C \ge 0$ **and** $\bigwedge r\ z.\ r > 0 \Longrightarrow r < pi \Longrightarrow z \in sphere\ 0\ r \Longrightarrow norm\ (f\ s\ z) \le C * r\ powr\ (-Re\ s - 1)$
proof −
have $2 * pi > 0$ **by** *auto*
have *nz*: $1 - exp\ z \ne 0$ **if** $z \in ball\ 0\ (2 * pi) - \{0\}$ **for** $z :: complex$
proof
assume $1 - exp\ z = 0$
then obtain *n* **where** $z = 2 * pi * of\text{-}int\ n * \mathrm{i}$

by (*auto simp*: *exp-eq-1 complex-eq-iff* [*of z*])
moreover have |*real-of-int n*| < *1* ⟷ *n* = *0* **by** *linarith*
ultimately show *False* **using** *that* **by** (*auto simp*: *norm-mult*)
qed

have *ev*: *eventually* (λ*z*::*complex*. *1* − *exp z* ≠ *0*) (*at 0*)
using *eventually-at-ball′*[*OF* ‹*2* ∗ *pi* > *0*›] **by** *eventually-elim* (*use nz* **in** *auto*)
have [*simp*]: *subdegree* (*1* − *fps-exp* (*1* :: *complex*)) = *1*
by (*intro subdegreeI*) *auto*
hence (λ*z*. *exp* (*a* ∗ *z*) ∗ (*if z* = *0 then* −*1 else z* / (*1* − *exp z* :: *complex*)))
has-fps-expansion fps-exp a ∗ (*fps-X* / (*fps-const 1* − *fps-exp 1*))
by (*auto intro*!: *fps-expansion-intros*)
hence (λ*z*::*complex*. *exp* (*a* ∗ *z*) ∗ (*if z* = *0 then* −*1 else z* / (*1* − *exp z*))) ∈ *O*[*at 0*](λ*z*. *1*)
using *continuous-imp-bigo-1 has-fps-expansion-imp-continuous* **by** *blast*
also have *?this* ⟷ (λ*z*::*complex*. *exp* (*a* ∗ *z*) ∗ (*z* / (*1* − *exp z*))) ∈ *O*[*at 0*](λ*z*. *1*)
by (*intro landau-o.big.in-cong eventually-mono*[*OF ev*]) *auto*
finally have ∃ *g*. *g holomorphic-on ball 0* (*2* ∗ *pi*) ∧
(∀ *z*∈*ball 0* (*2* ∗ *pi*) − {*0*}. *g z* = *exp* (*of-real a* ∗ *z*) ∗ (*z* / (*1* − *exp z*)))
using *nz* **by** (*intro holomorphic-on-extend holomorphic-intros*) *auto*
then guess *g* **by** (*elim exE conjE*) **note** *g* = *this*
hence *continuous-on* (*ball 0* (*2* ∗ *pi*)) *g*
by (*auto dest*: *holomorphic-on-imp-continuous-on*)
hence *continuous-on* (*cball 0 pi*) *g*
by (*rule continuous-on-subset*) (*subst cball-subset-ball-iff*, *use pi-gt-zero* **in** *auto*)
hence *compact* (*g* ‘ *cball 0 pi*) **by** (*intro compact-continuous-image*) *auto*
hence *bounded* (*g* ‘ *cball 0 pi*) **by** (*auto simp*: *compact-imp-bounded*)
then obtain *C* **where** *C*: ∀ *x*∈*cball 0 pi*. *norm* (*g x*) ≤ *C* **by** (*auto simp*: *bounded-iff*)

{
fix *r* :: *real* **assume** *r*: *r* > *0 r* < *pi*
fix *z* :: *complex* **assume** *z*: *z* ∈ *sphere 0 r*
define *x* **where** *x* = (*if Arg z* ≤ −*pi* / *2 then Arg z* + *2* ∗ *pi else Arg z*)
have *exp* (i ∗ (*2* ∗ *pi*)) = *1* **by** (*simp add*: *exp-eq-polar*)
with *z* **have** *z* = *r* ∗ *exp* (i ∗ *x*) **using** *r pi-gt-zero Arg-eq*[*of z*]
by (*auto simp*: *x-def exp-add distrib-left*)
have *x* > − *pi* / *2 x* ≤ *3* / *2* ∗ *pi* **using** *Arg-le-pi*[*of z*] *mpi-less-Arg*[*of z*]
by (*auto simp*: *x-def*)
note *x* = ‹*z* = *r* ∗ *exp* (i ∗ *x*)› *this*

from *x r* **have** *z′*: *z* ∈ *cball 0 pi* − {*0*}
using *pi-gt3* **by** (*auto simp*: *norm-mult*)
also have *cball 0 pi* ⊆ *ball* (*0*::*complex*) (*2* ∗ *pi*)
by (*subst cball-subset-ball-iff*) (*use pi-gt-zero* **in** *auto*)
hence *cball 0 pi* − {*0*} ⊆ *ball 0* (*2* ∗ *pi*) − {*0*::*complex*} **by** *blast*
finally have *z′′*: *z* ∈ *ball 0* (*2* ∗ *pi*) − {*0*} **.**

hence *bound*: *norm* (*exp* (*a* ∗ *z*) ∗ (*z* / (*1* − *exp z*))) ≤ *C* **using** *C* **and** *g* **and** *z′*
by *force*

have *exp z* ≠ *1* **using** *nz z″* **by** *auto*
with *bound z″* **have** *bound′*: *norm* (*exp* (*a* ∗ *z*) / (*1* − *exp z*)) ≤ *C* / *norm z*
by (*simp add*: *norm-divide field-simps norm-mult*)

have *Ln′ z* = *of-real* (*ln r*) + *Ln′* (*exp* (i ∗ *of-real x*))
using *x r* **by** (*simp add*: *Ln′-times-of-real*)
also have *exp* (i ∗ *pi* / *2*) = i
by (*simp add*: *exp-eq-polar*)
hence *Ln′* (*exp* (i ∗ *of-real x*)) = *Ln* (*exp* (i ∗ *of-real* (*x* − *pi* / *2*))) + i ∗ *pi* / *2*
by (*simp add*: *algebra-simps Ln′-def exp-diff*)
also have . . . = i ∗ *x*
using *x pi-gt3* **by** (*subst Ln-exp*) (*auto simp*: *algebra-simps*)
finally have *norm* (*exp* (−*Ln′ z* ∗ *s*)) = *exp* (*x* ∗ *Im s* − *ln r* ∗ *Re s*)
by *simp*
also {
have *x* ∗ *Im s* ≤ |*x* ∗ *Im s*| **by** (*rule abs-ge-self*)
also have . . . ≤ (*3*/*2* ∗ *pi*) ∗ |*Im s*| **unfolding** *abs-mult* **using** *x*
by (*intro mult-right-mono*) *auto*
finally have *exp* (*x* ∗ *Im s* − *ln r* ∗ *Re s*) ≤ *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*| − *ln r* ∗ *Re s*) **by** *simp*
}
finally have *norm* (*exp* (−*Ln′ z* ∗ *s*) ∗ (*exp* (*a* ∗ *z*) / (*1* − *exp z*))) ≤
exp (*3* / *2* ∗ *pi* ∗ |*Im s*| − *ln r* ∗ *Re s*) ∗ (*C* / *norm z*)
unfolding *norm-mult*[*of exp t* **for** *t*] **by** (*intro mult-mono bound′*) *simp-all*
also have *norm z* = *r* **using** ‹*r* > *0*› **by** (*simp add*: *x norm-mult*)
also have *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*| − *ln r* ∗ *Re s*) = *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*|) ∗ *r powr* (−*Re s*)
using *r* **by** (*simp add*: *exp-diff powr-def exp-minus inverse-eq-divide*)
finally have *norm* (*f s z*) ≤ *C* ∗ *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*|) ∗ *r powr* (−*Re s* − *1*) **using** *r*
by (*simp add*: *f-def exp-diff exp-minus field-simps powr-diff*)
also have . . . ≤ *max 0* (*C* ∗ *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*|)) ∗ *r powr* (−*Re s* − *1*)
by (*intro mult-right-mono max.coboundedI2*) *auto*
finally have *norm* (*f s z*) ≤ . . . .
}
with *that*[*of max 0* (*C* ∗ *exp* (*3* / *2* ∗ *pi* ∗ |*Im s*|))] **show** *?thesis* **by** *auto*
qed

We can now relate the integral along a partial Hankel contour that is cut off at $-\pi$ to $\zeta(1-s,a)/\Gamma(s)$.

lemma *rGamma-hurwitz-zeta-eq-contour-integral*:
fixes *s* :: *complex* **and** *r* :: *real*
assumes *s* ≠ *0* **and** *r*: *r* ∈ {*1*..<*2*} **and** *a*: *a* > *0*
defines *err1* ≡ (λ*s r*. *contour-integral* (*part-circlepath 0 r pi 0*) (*f s*))

defines *err2* ≡ (λ*s r*. *cnj* (*contour-integral* (*part-circlepath 0 r pi 0*) (*f* (*cnj s*))))
shows *2* ∗ i ∗ *pi* ∗ *rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*) =
err2 s r − *err1 s r* + *2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..}. *g s x*)
(**is** *?f s* = *?g s*)
proof (*rule analytic-continuation-open*[**where** *f* = *?f*])
fix *s* :: *complex* **assume** *s*: *s* ∈ {*s*. *Re s* < *0*}

— We first show that the integrals along the Hankel contour cut off at $-\pi$ all have the same value, no matter what the radius of the circle is (as long as it is small enough). We call this value C.
This argument could be done by a homotopy argument, but it is easier to simply re-use the above result about the contour integral along the annulus where we fix the radius of the outer circle to π.
define *C* **where** *C* = −*contour-integral* (*part-circlepath 0 pi 0 pi*) (*f s*) +
cnj (*contour-integral* (*part-circlepath 0 pi 0 pi*) (*f* (*cnj s*)))
have *integrable*: *set-integrable lborel A* (*g s*)
if *A* ∈ *sets lborel A* ⊆ {*0*<..} **for** *A*
proof (*rule set-integrable-subset*)
show *set-integrable lborel* {*0*<..} (*g s*)
using *Gamma-times-hurwitz-zeta-integrable*[*of 1* − *s a*] *s a*
by (*simp add*: *g-def exp-of-real exp-minus integrable-completion set-integrable-def*)
qed (*insert that*, *auto*)

{
fix *r'* :: *real* **assume** *r'*: *r'* ∈ {*0*<..<*2*}
from *hurwitz-formula-integral-semiannulus*(*2*)[*of r' s 0*] **and** *r'*
have *f s contour-integrable-on part-circlepath 0 r' pi 0*
by (*auto simp*: *hankel-semiannulus-def add-ac*)
} **note** *integrable-circle* = *this*
{
fix *r'* :: *real* **assume** *r'*: *r'* ∈ {*0*<..<*2*}
from *hurwitz-formula-integral-semiannulus*(*2*)[*of r' cnj s 0*] **and** *r'*
have *f* (*cnj s*) *contour-integrable-on part-circlepath 0 r' pi 0*
by (*auto simp*: *hankel-semiannulus-def add-ac*)
} **note** *integrable-circle'* = *this*

have *eq*: −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..*pi*}. *g s x*) + (*err1 s r* − *err2 s r*) = *C*
if *r*: *r* ∈ {*0*<..<*2*} **for** *r* :: *real*
proof −
have *eq1*: *integral* {*r*..*pi*} (λ*x*. *cnj* (*x powr* − *cnj s*) ∗ (*exp* (− (*a* ∗ *x*))) / (*1* − (*exp* (− *x*)))) =
integral {*r*..*pi*} (*g s*) **using** *r*
by (*intro integral-cong*) (*auto simp*: *cnj-powr g-def exp-of-real exp-minus*)
have *eq2*: *integral* {*r*..*pi*} (λ*x*. *cnj* (*x powr* − *cnj s*) ∗ (*exp* (*a* ∗ *x*)) / (*1* − (*exp x*))) =
integral {*r*..*pi*} (λ*x*. *x powr* − *s* ∗ (*exp* (*a* ∗ *x*)) / (*1* − (*exp x*))) **using** *r*
by (*intro integral-cong*) (*auto simp*: *cnj-powr*)

from *hurwitz-formula-integral-semiannulus*(*1*)[*of r s 0*] *hurwitz-formula-integral-semiannulus*(*1*)[*of r cnj s 0*]
have *exp* (−i∗*pi* ∗ *s*) ∗
integral {*r*..*real* (*2*∗*0*+*1*) ∗ *pi*} (*g s*) +
integral {*r*..*real* (*2*∗*0*+*1*) ∗ *pi*} (λ*x*. *x powr* −*s* ∗ *exp* (*a* ∗ *x*) / (*1* − *exp x*)) +
contour-integral (*part-circlepath 0* (*real* (*2* ∗ *0* + *1*) ∗ *pi*) *0 pi*) (*f s*) +
contour-integral (*part-circlepath 0 r pi 0*) (*f s*) − *cnj* (
exp (−i∗*pi* ∗ *cnj s*) ∗
integral {*r*..*real* (*2*∗*0*+*1*) ∗ *pi*} (λ*x*. *x powr* − *cnj s* ∗ *exp* (−*a*∗*x*) / (*1* − *exp* (−*x*))) +
integral {*r*..*real* (*2*∗*0*+*1*) ∗ *pi*} (λ*x*. *x powr* − *cnj s* ∗ *exp* (*a*∗*x*) / (*1* − *exp x*)) +
contour-integral (*part-circlepath 0* (*real* (*2* ∗ *0* + *1*) ∗ *pi*) *0 pi*) (*f* (*cnj s*)) +
contour-integral (*part-circlepath 0 r pi 0*) (*f* (*cnj s*))) = *0* (**is** *?lhs* = -)
unfolding *g-def* **using** *r* **by** (*subst* (*1 2*) *hurwitz-formula-integral-semiannulus*) *auto*
also have *?lhs* = −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ *integral* {*r*..*pi*} (*g s*) + *err1 s r* − *err2 s r* − *C*
using *eq1 eq2*
by (*auto simp*: *integral-cnj exp-cnj err1-def err2-def sin-exp-eq algebra-simps C-def*)
also have *integral* {*r*..*pi*} (*g s*) = (*CLBINT x*:{*r*..*pi*}. *g s x*) **using** *r*
by (*intro set-borel-integral-eq-integral*(*2*) [*symmetric*] *integrable*) *auto*
finally show −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..*pi*}. *g s x*) + (*err1 s r* − *err2 s r*) = *C*
by (*simp add*: *algebra-simps*)
qed

— Next, compute the value of C by letting the radius tend to 0 so that the contribution of the circle vanishes.
have ((λ*r*. −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..*pi*}. *g s x*) + (*err1 s r* − *err2 s r*)) ⟶
−*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*0*<..*pi*}. *g s x*) + *0*) (*at-right 0*)
proof (*intro tendsto-intros tendsto-set-lebesgue-integral-at-right integrable*)
from *hurwitz-formula-bound2*[*of s*] **guess** *C1* **. note** *C1* = *this*
from *hurwitz-formula-bound2*[*of cnj s*] **guess** *C2* **. note** *C2* = *this*
have *ev*: *eventually* (λ*r*::*real*. *r* ∈ {*0*<..<*2*}) (*at-right 0*)
by (*intro eventually-at-right-real*) *auto*
show ((λ*r*. *err1 s r* − *err2 s r*) ⟶ *0*) (*at-right 0*)
proof (*rule Lim-null-comparison*[*OF eventually-mono*[*OF ev*]])
fix *r* :: *real* **assume** *r*: *r* ∈ {*0*<..<*2*}
have *norm* (*err1 s r* − *err2 s r*) ≤ *norm* (*err1 s r*) + *norm* (*err2 s r*)
by (*rule norm-triangle-ineq4*)
also have *norm* (*err1 s r*) ≤ *C1* ∗ *r powr* (− *Re s* − *1*) ∗ *r* ∗ |*0* − *pi*|
unfolding *err1-def* **using** *C1*(*1*) *C1*(*2*)[*of r*] *pi-gt3 integrable-circle*[*of r*]
path-image-part-circlepath-subset'[*of r 0 pi 0*] *r*

by (*intro contour-integral-bound-part-circlepath*) *auto*
also have ... = *C1* ∗ *r powr* (−*Re s*) ∗ *pi* **using** *r*
by (*simp add*: *powr-diff field-simps*)
also have *norm* (*err2 s r*) ≤ *C2* ∗ *r powr* (− *Re s* − *1*) ∗ *r* ∗ |*0* − *pi*|
unfolding *err2-def complex-mod-cnj* **using** *C2*(*1*) *C2*(*2*)[*of r*] *r*
pi-gt3 integrable-circle′[*of r*] *path-image-part-circlepath-subset′*[*of r 0 pi 0*]
by (*intro contour-integral-bound-part-circlepath*) *auto*
also have ... = *C2* ∗ *r powr* (−*Re s*) ∗ *pi* **using** *r*
by (*simp add*: *powr-diff field-simps*)
also have *C1* ∗ *r powr* (−*Re s*) ∗ *pi* + *C2* ∗ *r powr* (−*Re s*) ∗ *pi* =
(*C1* + *C2*) ∗ *pi* ∗ *r powr* (−*Re s*) **by** (*simp add*: *algebra-simps*)
finally show *norm* (*err1 s r* − *err2 s r*) ≤ (*C1* + *C2*) ∗ *pi* ∗ *r powr* − *Re s*
by *simp*
next
show ((λ*x*. (*C1* + *C2*) ∗ *pi* ∗ *x powr* − *Re s*) ⟶ *0*) (*at-right 0*) **using** *s*
by (*auto intro*!: *tendsto-eq-intros simp*: *eventually-at exI*[*of - 1*])
qed
qed *auto*
moreover have *eventually* (λ*r*::*real*. *r* ∈ {*0*<..<*2*}) (*at-right 0*)
by (*intro eventually-at-right-real*) *auto*
hence *eventually* (λ*r*. −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..*pi*}. *g s x*) +
(*err1 s r* − *err2 s r*) = *C*) (*at-right 0*) **by** *eventually-elim* (*use eq* **in** *auto*)
hence ((λ*r*. −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..*pi*}. *g s x*) + (*err1 s r* − *err2 s r*)) ⟶ *C*)
(*at-right 0*) **by** (*rule tendsto-eventually*)
ultimately have [*simp*]: *C* = −*2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*0*<..*pi*}. *g s x*)
using *tendsto-unique* **by** *force*

— We now rearrange everything and obtain the result.
have *2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ ((*CLBINT x*:{*0*<..*pi*}. *g s x*) − (*CLBINT x*:{*r*..*pi*}. *g s x*)) =
err2 s r − *err1 s r*
using *eq*[*of r*] *r* **by** (*simp add*: *algebra-simps*)
also have {*0*<..*pi*} = {*0*<..<*r*} ∪ {*r*..*pi*} **using** *r pi-gt3* **by** *auto*
also have (*CLBINT x*:... *g s x*) − (*CLBINT x*:{*r*..*pi*}. *g s x*) = (*CLBINT x*:{*0*<..<*r*}. *g s x*)
using *r pi-gt3* **by** (*subst set-integral-Un*[*OF - integrable integrable*]) *auto*
also have (*CLBINT x*:{*0*<..<*r*}. *g s x*) =
(*CLBINT x*:{*0*<..<*r*} ∪ {*r*..}. *g s x*) − (*CLBINT x*:{*r*..}. *g s x*)
using *r pi-gt3* **by** (*subst set-integral-Un*[*OF - integrable integrable*]) *auto*
also have {*0*<..<*r*} ∪ {*r*..} = {*0*<..} **using** *r* **by** *auto*
also have (*CLBINT x*:{*0*<..}. *g s x*) = *Gamma* (*1* − *s*) ∗ *hurwitz-zeta a* (*1* − *s*)
using *Gamma-times-hurwitz-zeta-integral*[*of 1* − *s a*] *s a*
by (*simp add*: *g-def exp-of-real exp-minus integral-completion set-lebesgue-integral-def*)
finally have *2* ∗ i ∗ (*sin* (*pi* ∗ *s*) ∗ *Gamma* (*1* − *s*)) ∗ *hurwitz-zeta a* (*1* − *s*) =
err2 s r − *err1 s r* + *2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..}. *g s x*)
by (*simp add*: *algebra-simps*)

also have *sin* (*pi* ∗ *s*) ∗ *Gamma* (*1* − *s*) = *pi* ∗ *rGamma s*
proof (*cases s* ∈ $\mathbb{Z}$)
case *False*
with *Gamma-reflection-complex*[*of s*] **show** *?thesis*
by (*auto simp*: *divide-simps sin-eq-0 Ints-def rGamma-inverse-Gamma mult-ac split*: *if-splits*)
next
case *True*
with *s* **have** *rGamma s* = *0*
by (*auto simp*: *rGamma-eq-zero-iff nonpos-Ints-def Ints-def*)
moreover from *True* **have** *sin* (*pi* ∗ *s*) = *0*
by (*subst sin-eq-0*) (*auto elim*!: *Ints-cases*)
ultimately show *?thesis* **by** *simp*
qed
finally show *2* ∗ i ∗ *pi* ∗ *rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*) =
err2 s r − *err1 s r* + *2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..}. *g s x*)
by (*simp add*: *mult-ac*)
next
— By analytic continuation, we lift the result to the case of any non-zero *s*.
show (λ*s*. *2* ∗ i ∗ *pi* ∗ *rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*)) *holomorphic-on* −{*0*} **using** *a*
by (*auto intro*!: *holomorphic-intros*)
show (λ*s*. *err2 s r* − *err1 s r* + *2* ∗ i ∗ *sin* (*pi* ∗ *s*) ∗ (*CLBINT x*:{*r*..}. *g s x*))
holomorphic-on −{*0*}
proof (*intro holomorphic-intros*)
have (λ*s*. *err2 s r*) = (λ*s*. − *cnj* (*integral* {*0*..*pi*} (*h r* (*cnj s*)))) **using** *r*
by (*simp add*: *err2-def contour-integral-part-circlepath-reverse′*
contour-integral-part-circlepath-h)
also have (λ*s*. − *cnj* (*integral* {*0*..*pi*} (*h r* (*cnj s*)))) =
(λ*s*. (*integral* {*0*..*pi*} (λ*x*. *h r s* (−*x*)))) **using** *r*
by (*simp add*: *integral-cnj h-def exp-cnj cis-cnj Ln-Reals-eq*)
also have ... = (λ*s*. *integral* {−*pi*..*0*} (*h r s*))
by (*subst Henstock-Kurzweil-Integration.integral-reflect-real* [*symmetric*]) *simp*
finally have (λ*s*. *err2 s r*) = ... **.**
moreover have (λ*s*. *integral* {−*pi*..*0*} (*h r s*)) *holomorphic-on* −{*0*}
using *r* **by** (*intro integral-h-holomorphic*) *auto*
ultimately show (λ*s*. *err2 s r*) *holomorphic-on* −{*0*} **by** *simp*
next
have (λ*s*. − *integral* {*0*..*pi*} (*h r s*)) *holomorphic-on* −{*0*} **using** *r*
by (*intro holomorphic-intros integral-h-holomorphic*) *auto*
also have (λ*s*. − *integral* {*0*..*pi*} (*h r s*)) = (λ*s*. *err1 s r*)
unfolding *err1-def* **using** *r*
by (*simp add*: *contour-integral-part-circlepath-reverse′ contour-integral-part-circlepath-h*)
finally show (λ*s*. *err1 s r*) *holomorphic-on* −{*0*} **.**
next
show (λ*s*. *CLBINT x*:{*r*..}. *g s x*) *holomorphic-on* −{*0*}
proof (*rule holomorphic-on-balls-imp-entire′*)
fix *R* :: *real*
have *eventually* (λ*b*. *b* > *r*) *at-top* **by** (*rule eventually-gt-at-top*)

hence *1*: *eventually* (*λb. continuous-on* (*cball 0 R*) (*λs. CLBINT x*:{*r..b*}. *g s x*) ∧
(*λs. CLBINT x*:{*r..b*}. *g s x*) *holomorphic-on ball 0 R*) *at-top*
proof *eventually-elim*
case (*elim b*)
have *integrable*: *set-integrable lborel* {*r..b*} (*g s*) **for** *s* **unfolding** *g-def* **using** *r*
by (*intro borel-integrable-atLeastAtMost′ continuous-intros*) *auto*
have (*λs. integral* {*r..b*} (*g s*)) *holomorphic-on UNIV* **using** *r*
by (*intro integral-g-holomorphic*) *auto*
also have (*λs. integral* {*r..b*} (*g s*)) = (*λs. CLBINT x*:{*r..b*}. *g s x*)
by (*intro ext set-borel-integral-eq-integral*(*2*)[*symmetric*] *integrable*)
finally have ... *holomorphic-on UNIV* **.**
thus *?case* **by** (*auto intro*!: *holomorphic-on-imp-continuous-on*)
qed

have *2*: *uniform-limit* (*cball 0 R*) (*λb s. CLBINT x*:{*r..b*}. *g s x*)
(*λs. CLBINT x*:{*r..*}. *g s x*) *at-top*
proof (*rule uniform-limit-set-lebesgue-integral-at-top*)
fix *s* :: *complex* **and** *x* :: *real*
assume *s*: *s* ∈ *cball 0 R* **and** *x*: *x* ≥ *r*
have *norm* (*g s x*) = *x powr* −*Re s* ∗ *exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*)) **using** *x r*
by (*simp add*: *g-def norm-mult norm-divide in-Reals-norm norm-powr-real-powr*)
also have ... ≤ *x powr R* ∗ *exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*)) **using** *r s x abs-Re-le-cmod*[*of s*]
by (*intro mult-right-mono divide-right-mono powr-mono*) *auto*
finally show *norm* (*g s x*) ≤ *x powr R* ∗ *exp* (− *a* ∗ *x*) / (*1* − *exp* (− *x*)) **.**
next
show *set-integrable lborel* {*r..*} (*λx. x powr R* ∗ *exp* (−*a* ∗ *x*) / (*1* − *exp* (−*x*)))
using *r a* **by** (*intro set-integrable-Gamma-hurwitz-aux2-real*) *auto*
qed (*simp-all add*: *set-borel-measurable-def g-def*)

show (*λs. CLBINT x*:{*r..*}. *g s x*) *holomorphic-on ball 0 R*
using *holomorphic-uniform-limit*[*OF 1 2*] **by** *auto*
qed
qed
qed (*insert* ‹*s* ≠ *0*›,
auto simp: *connected-punctured-universe open-halfspace-Re-lt intro*: *exI*[*of* - −*1*])

Finally, we obtain Hurwitz's formula by letting the radius of the outer circle tend to ∞.

lemma *hurwitz-zeta-formula-aux*:
fixes *s* :: *complex*
assumes *s*: *Re s* > *1*
shows *rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*) = (*2* ∗ *pi*) *powr* −*s* ∗

(i *powr* $(-s) * F\ a\ s$ + i *powr* $s * F\ (-a)\ s$)
proof −
from *s* **have** [*simp*]: $s \neq 0$ **by** *auto*
define *r* **where** $r = (1 :: real)$
have *r*: $r \in \{0<..<2\}$ **by** (*simp add*: *r-def*)
define *R* **where** $R = (\lambda n.\ real\ (2 * n + 1) * pi)$
define *bigc* **where** *bigc* = (λ*n*. *contour-integral* (*part-circlepath 0* (*R n*) *0 pi*) (*f s*) −
cnj (*contour-integral* (*part-circlepath 0* (*R n*) *0 pi*) (*f* (*cnj s*))))
define *smallc* **where** *smallc* = *contour-integral* (*part-circlepath 0 r pi 0*) (*f s*) −
cnj (*contour-integral* (*part-circlepath 0 r pi 0*) (*f* (*cnj s*)))
define *I* **where** *I* = (λ*n*. *CLBINT x*:{*r*..*R n*}. *g s x*)

define *F1* **and** *F2* **where**
F1 = (λ*n*. *exp* $(-s * pi *$ i $/\ 2) * (\sum k\in\{0<..n\}.\ exp\ (2 * real\ k * pi * a *$ i$) * k$ *powr* $(-s)))$
F2 = (λ*n*. *exp* $(s * pi *$ i $/\ 2) * (\sum k\in\{0<..n\}.\ exp\ (2 * real\ k * pi * (-a) *$ i$) * k$ *powr* $(-s)))$

have *R*: $R\ n \geq pi$ **for** *n* **using** *r* **by** (*auto simp*: *R-def field-simps*)
have [*simp*]: $\neg(pi \leq 0)$ **using** *pi-gt-zero* **by** *linarith*

have *integrable*: *set-integrable lborel A* (*g s*)
if $A \in$ *sets lborel* $A \subseteq \{r..\}$ **for** *A*
proof −
have *set-integrable lborel* {*r*..} (*g s*)
using *set-integrable-Gamma-hurwitz-aux2*[*of r a s*] *a r*
by (*simp add*: *g-def exp-of-real exp-minus*)
thus *?thesis* **by** (*rule set-integrable-subset*) (*use that* **in** *auto*)
qed

{
fix *n* :: *nat*
from *hurwitz-formula-integral-semiannulus*(*2*)[*of r s n*] **and** *r R*[*of n*]
have *f s contour-integrable-on part-circlepath 0* (*R n*) *0 pi*
by (*auto simp*: *hankel-semiannulus-def R-def add-ac*)
} **note** *integrable-circle* = *this*
{
fix *n* :: *nat*
from *hurwitz-formula-integral-semiannulus*(*2*)[*of r cnj s n*] **and** *r R*[*of n*]
have *f* (*cnj s*) *contour-integrable-on part-circlepath 0* (*R n*) *0 pi*
by (*auto simp*: *hankel-semiannulus-def R-def add-ac*)
} **note** *integrable-circle′* = *this*

{
fix *n* :: *nat*
have (*exp* (−i ∗ *pi* ∗ *s*) ∗ *integral* {*r*..*R n*} (*g s*) +

integral {*r*..*R n*} (λ*x*. *x powr* (−*s*) ∗ *exp* (*a* ∗ *x*) / (*1* − *exp x*)) +
contour-integral (*part-circlepath 0* (*R n*) *0 pi*) (*f s*) +
contour-integral (*part-circlepath 0 r pi 0*) (*f s*)) − *cnj* (
exp (−i ∗ *pi* ∗ *cnj s*) ∗ *integral* {*r*..*R n*} (*g* (*cnj s*)) +
integral {*r*..*R n*} (λ*x*. *x powr* (−*cnj s*) ∗ *exp* (*a* ∗ *x*) / (*1* − *exp x*)) +
contour-integral (*part-circlepath 0* (*R n*) *0 pi*) (*f* (*cnj s*)) +
contour-integral (*part-circlepath 0 r pi 0*) (*f* (*cnj s*)))
= −*2* ∗ *pi* ∗ i ∗ *exp* (− *s* ∗ *of-real pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}. *Res s k*) −
cnj (−*2* ∗ *pi* ∗ i ∗ *exp* (− *cnj s* ∗ *of-real pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}.
Res (*cnj s*) *k*))
(**is** *?lhs* = *?rhs*) **unfolding** *R-def g-def* **using** *r*
by (*subst* (*1 2*) *hurwitz-formula-integral-semiannulus*) *auto*
also have *?rhs* = −*2* ∗ *pi* ∗ i ∗ (*exp* (− *s* ∗ *pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}. *Res*
s k) +
exp (*s* ∗ *pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}. *cnj* (*Res* (*cnj*
s) *k*)))
by (*simp add*: *exp-cnj sum.distrib algebra-simps sum-distrib-left sum-distrib-right*
sum-negf)
also have (∑ *k*∈{*0*<..*n*}. *Res s k*) =
(*2* ∗ *pi*) *powr* (−*s*) ∗ (∑ *k*∈{*0*<..*n*}. *exp* (*2* ∗ *k* ∗ *pi* ∗ *a* ∗ i) ∗ *k*
powr (−*s*))
(**is** - = *?S1*) **by** (*simp add*: *Res-def powr-times-real algebra-simps sum-distrib-left*)
also have (∑ *k*∈{*0*<..*n*}. *cnj* (*Res* (*cnj s*) *k*)) =
(*2* ∗ *pi*) *powr* (−*s*) ∗ (∑ *k*∈{*0*<..*n*}. *exp* (−*2* ∗ *k* ∗ *pi* ∗ *a* ∗ i) ∗ *k*
powr (−*s*))
by (*simp add*: *Res-def cnj-powr powr-times-real algebra-simps exp-cnj sum-distrib-left*)
also have *exp* (−*s* ∗ *pi* ∗ i / *2*) ∗ *?S1* + *exp* (*s* ∗ *pi* ∗ i / *2*) ∗ ... =
(*2* ∗ *pi*) *powr* (−*s*) ∗
(*exp* (−*s* ∗ *pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}. *exp* (*2* ∗ *k* ∗ *pi* ∗ *a* ∗ i) ∗
k powr (−*s*)) +
exp (*s* ∗ *pi* ∗ i / *2*) ∗ (∑ *k*∈{*0*<..*n*}. *exp* (−*2* ∗ *k* ∗ *pi* ∗ *a* ∗ i) ∗
k powr (−*s*)))
by (*simp add*: *algebra-simps*)
also have *1*: *integral* {*r*..*R n*} (*g s*) = *I n* **unfolding** *I-def*
by (*intro set-borel-integral-eq-integral*(*2*) [*symmetric*] *integrable*) *auto*
have *2*: *cnj* (*integral* {*r*..*R n*} (*g* (*cnj s*))) = *integral* {*r*..*R n*} (*g s*) **using** *r*
unfolding *integral-cnj* **by** (*intro integral-cong*) (*auto simp*: *g-def cnj-powr*)
have *3*: *integral* {*r*..*R n*} (λ*x*. *exp* (*x* ∗ *a*) ∗ *cnj* (*x powr* − *cnj s*) / (*1* − *exp*
x)) =
integral {*r*..*R n*} (λ*x*. *exp* (*x* ∗ *a*) ∗ *of-real x powr* − *s* / (*1* − *exp x*))
unfolding *I-def g-def* **using** *r R*[*of n*] **by** (*intro integral-cong*; *force simp*:
cnj-powr)+
from *1 2 3* **have** *?lhs* = (*exp* (−i ∗ *s* ∗ *pi*) − *exp* (i ∗ *s* ∗ *pi*)) ∗ *I n* + *bigc n*
+ *smallc*
by (*simp add*: *integral-cnj cnj-powr algebra-simps exp-cnj*
bigc-def smallc-def g-def)
also have *exp* (−i ∗ *s* ∗ *pi*) − *exp* (i ∗ *s* ∗ *pi*) = −*2* ∗ i ∗ *sin* (*s* ∗ *pi*)
by (*simp add*: *sin-exp-eq′ algebra-simps*)
finally have (− *2* ∗ i ∗ *sin* (*s* ∗ *pi*) ∗ *I n* + *smallc*) + *bigc n* =

$-2 * \mathrm{i} * pi * (2 * pi)$ *powr* $(-s) * (F1\ n + F2\ n)$
by (*simp add*: *F1-F2-def algebra-simps*)
} **note** $eq = this$

have $(\lambda n. - 2 * \mathrm{i} * sin\ (s * pi) * I\ n + smallc + bigc\ n) \longrightarrow$
$(-2 * \mathrm{i} * sin\ (s * pi)) * (CLBINT\ x{:}\{r..\}.\ g\ s\ x) + smallc + 0$
unfolding *I-def*
proof (*intro tendsto-intros filterlim-compose*[*OF tendsto-set-lebesgue-integral-at-top*] *integrable*)
show *filterlim R at-top sequentially* **unfolding** *R-def*
by (*intro filterlim-at-top-mult-tendsto-pos*[*OF tendsto-const*] *pi-gt-zero*
filterlim-compose[*OF filterlim-real-sequentially*] *filterlim-subseq*)
(*auto simp*: *strict-mono-Suc-iff*)

from *hurwitz-formula-bound1*[*OF pi-gt-zero*] **guess** C **. note** $C = this$
define D **where** $D = C * exp\ (3\ /\ 2 * pi * |Im\ s|)$
from ‹$C \geq 0$› **have** $D \geq 0$ **by** (*simp add*: *D-def*)
show $bigc \longrightarrow 0$
proof (*rule Lim-null-comparison*[*OF always-eventually*[*OF allI*]])
fix $n :: nat$
have *bound*: $norm\ (f\ s'\ z) \leq D * R\ n$ *powr* $(-Re\ s')$
if z: $z \in sphere\ 0\ (R\ n)$ $Re\ s' = Re\ s$ $|Im\ s'| = |Im\ s|$ **for** $z\ s'$
proof −
from z **and** r R[*of n*] **have** [*simp*]: $z \neq 0$ **by** *auto*
have *not-in-ball*: $z \notin ball\ (2 * m * pi * \mathrm{i})\ pi$ **for** $m :: int$
proof −
have $dist\ z\ (2 * m * pi * \mathrm{i}) \geq |dist\ z\ 0 - dist\ 0\ (2 * m * pi * \mathrm{i})|$
by (*rule abs-dist-diff-le*)
also have $dist\ 0\ (2 * m * pi * \mathrm{i}) = 2 * |m| * pi$
by (*simp add*: *norm-mult*)
also from z **have** $dist\ z\ 0 = R\ n$ **by** *simp*
also have $R\ n - 2 * |m| * pi = (int\ (2 * n + 1) - 2 * |m|) * pi$
by (*simp add*: *R-def algebra-simps*)
also have $|\ldots| = |int\ (2 * n + 1) - 2 * |m|| * pi$
by (*subst abs-mult*) *simp-all*
also have $|int\ (2 * n + 1) - 2 * |m|| \geq 1$ **by** *presburger*
hence $\ldots * pi \geq 1 * pi$ **by** (*intro mult-right-mono*) *auto*
finally show *?thesis* **by** (*simp add*: *dist-commute*)
qed

have $norm\ (f\ s'\ z) = norm\ (exp\ (-Ln'\ z * s')) * norm\ (exp\ (a * z)\ /\ (1 - exp\ z))$
by (*simp add*: *f-def exp-diff norm-mult norm-divide mult-ac exp-minus norm-inverse*
divide-simps del: *norm-exp-eq-Re*)
also have $\ldots \leq norm\ (exp\ (-Ln'\ z * s')) * C$ **using** *not-in-ball*
by (*intro mult-left-mono C*) *auto*
also have $norm\ (exp\ (-Ln'\ z * s')) =$

exp (*Im s'* ∗ (*Im* (*Ln* (− (i ∗ *z*))) + *pi* / *2*)) / *exp* (*Re s'* ∗ *ln* (*R n*))
using *z r R*[*of n*] *pi-gt-zero*
by (*simp add*: *Ln'-def norm-mult norm-divide exp-add exp-diff exp-minus norm-inverse algebra-simps inverse-eq-divide*)
also have … ≤ *exp* (*3*/*2* ∗ *pi* ∗ |*Im s'*|) / *exp* (*Re s'* ∗ *ln* (*R n*))
proof (*intro divide-right-mono*, *subst exp-le-cancel-iff*)
have *Im s'* ∗ (*Im* (*Ln* (− (i ∗ *z*))) + *pi* / *2*) ≤ |*Im s'* ∗ (*Im* (*Ln* (− (i ∗ *z*))) + *pi* / *2*)|
by (*rule abs-ge-self*)
also have … ≤ |*Im s'*| ∗ (*pi* + *pi* / *2*)
unfolding *abs-mult* **using** *mpi-less-Im-Ln*[*of* − (i ∗ *z*)] *Im-Ln-le-pi*[*of* − (i ∗ *z*)]
by (*intro mult-left-mono order.trans*[*OF abs-triangle-ineq*] *add-mono*) *auto*
finally show *Im s'* ∗ (*Im* (*Ln* (− (i ∗ *z*))) + *pi* / *2*) ≤ *3*/*2* ∗ *pi* ∗ |*Im s'*|
by (*simp add*: *algebra-simps*)
qed *auto*
also have *exp* (*Re s'* ∗ *ln* (*R n*)) = *R n powr Re s'*
using *r R*[*of n*] **by** (*auto simp*: *powr-def*)
finally show *norm* (*f s' z*) ≤ *D* ∗ *R n powr* (−*Re s'*) **using** ‹*C* ≥ *0*›
by (*simp add*: *that D-def powr-minus mult-right-mono mult-left-mono field-simps*)
qed

have *norm* (*bigc n*) ≤ *norm* (*contour-integral* (*part-circlepath 0* (*R n*) *0 pi*) (*f s*)) +
norm (*cnj* (*contour-integral* (*part-circlepath 0* (*R n*) *0 pi*) (*f* (*cnj s*))))
(**is** - ≤ *norm ?err1* + *norm ?err2*) **unfolding** *bigc-def* **by** (*rule norm-triangle-ineq4*)
also have *norm ?err1* ≤ *D* ∗ *R n powr* (−*Re s*) ∗ *R n* ∗ |*pi* − *0*|
using ‹*D* ≥ *0*› **and** *r R*[*of n*] **and** *pi-gt3* **and** *integrable-circle* **and**
path-image-part-circlepath-subset[*of 0 pi R n 0*] **and** *bound*[*of - s*]
by (*intro contour-integral-bound-part-circlepath*) *auto*
also have … = *D* ∗ *pi* ∗ *R n powr* (*1* − *Re s*) **using** *r R*[*of n*] *pi-gt3*
by (*simp add*: *powr-diff field-simps powr-minus*)
also have *norm ?err2* ≤ *D* ∗ *R n powr* (−*Re s*) ∗ *R n* ∗ |*pi* − *0*|
unfolding *complex-mod-cnj*
using ‹*D* ≥ *0*› **and** *r R*[*of n*] **and** *pi-gt3* **and** *integrable-circle'*[*of n*] **and**
path-image-part-circlepath-subset[*of 0 pi R n 0*] **and** *bound*[*of - cnj s*]
by (*intro contour-integral-bound-part-circlepath*) *auto*
also have … = *D* ∗ *pi* ∗ *R n powr* (*1* − *Re s*) **using** *r R*[*of n*] *pi-gt3*
by (*simp add*: *powr-diff field-simps powr-minus*)
finally show *norm* (*bigc n*) ≤ *2* ∗ *D* ∗ *pi* ∗ *R n powr* (*1* − *Re s*)
by *simp*
next
have *filterlim R at-top at-top* **by** *fact*
hence (λ*x*. *2* ∗ *D* ∗ *pi* ∗ *R x powr* (*1* − *Re s*)) ⟶ *2* ∗ *D* ∗ *pi* ∗ *0* **using** *s* **unfolding** *R-def*
by (*intro tendsto-intros tendsto-neg-powr*) *auto*

thus $(\lambda x.\ 2 * D * pi * R\ x\ powr\ (1 - Re\ s)) \longrightarrow 0$ **by** *simp*
qed
qed *auto*
also have $(\lambda n. - 2 * \mathrm{i} * sin\ (s * pi) * I\ n + smallc + bigc\ n) =$
$(\lambda n. -2 * \mathrm{i} * pi * (2 * pi)\ powr\ -s * (F1\ n + F2\ n))$ **by** (*subst eq*) *auto*
finally have ... $\longrightarrow (-2 * \mathrm{i} * sin\ (s * pi)) * (CLBINT\ x{:}\{r..\}.\ g\ s\ x) + smallc$ **by** *simp*

moreover have $(\lambda n. -2 * \mathrm{i} * pi * (2 * pi)\ powr\ -s * (F1\ n + F2\ n)) \longrightarrow$
$-2 * \mathrm{i} * pi * (2 * pi)\ powr\ -s *$
$(exp\ (-s * pi * \mathrm{i}\ /\ 2) * F\ a\ s + exp\ (s * pi * \mathrm{i}\ /\ 2) * F\ (-a)\ s)$
unfolding *F1-F2-def F-def* **using** *s* **by** (*intro tendsto-intros sum-tendsto-fds-perzeta*)
ultimately have $-2 * \mathrm{i} * pi * (2 * pi)\ powr\ -s *$
$(exp\ (-s * pi * \mathrm{i}\ /\ 2) * F\ a\ s + exp\ (s * pi * \mathrm{i}\ /\ 2) * F\ (-a)\ s) =$
$(-2 * \mathrm{i} * sin\ (s * pi)) * (CLBINT\ x{:}\{r..\}.\ g\ s\ x) + smallc$
by (*force intro*: *tendsto-unique*)
also have ... $= -2 * \mathrm{i} * pi * rGamma\ s * hurwitz\text{-}zeta\ a\ (1 - s)$ **using** *s r a*
using *rGamma-hurwitz-zeta-eq-contour-integral*[*of s r*]
by (*simp add*: *r-def smallc-def algebra-simps*)
also have $exp\ (- s * complex\text{-}of\text{-}real\ pi * \mathrm{i}\ /\ 2) = \mathrm{i}\ powr\ (-s)$
by (*simp add*: *powr-def field-simps*)
also have $exp\ (s * complex\text{-}of\text{-}real\ pi * \mathrm{i}\ /\ 2) = \mathrm{i}\ powr\ s$
by (*simp add*: *powr-def field-simps*)
finally show $rGamma\ s * hurwitz\text{-}zeta\ a\ (1 - s) = (2 * pi)\ powr\ -s *$
$(\mathrm{i}\ powr\ (-s) * F\ a\ s + \mathrm{i}\ powr\ s * F\ (-a)\ s)$ **by** *simp*
qed

end

We can now use Hurwitz's formula to prove the following nice formula that expresses the periodic zeta function in terms of the Hurwitz zeta function:

$$F(s,a) = (2\pi)^{s-1} i\Gamma(1-s)\left(i^{-s}\zeta(1-s,a) - i^{s}\zeta(1-s,1-a)\right)$$

This holds for all s with \mathfrak{R}$(s) > 0$ as long as $a \notin \mathbb{Z}$. For convenience, we move the Γ function to the left-hand side in order to avoid having to account for its poles.

lemma *perzeta-conv-hurwitz-zeta-aux*:
fixes $a :: real$ **and** $s :: complex$
assumes a: $a \in \{0{<}..{<}1\}$ **and** s: $Re\ s > 0$
shows $rGamma\ (1 - s) * eval\text{-}fds\ (fds\text{-}perzeta\ a)\ s = (2 * pi)\ powr\ (s - 1) * \mathrm{i} *$
$(\mathrm{i}\ powr\ -s * hurwitz\text{-}zeta\ a\ (1 - s) -$
$\mathrm{i}\ powr\ s * hurwitz\text{-}zeta\ (1 - a)\ (1 - s))$
(**is** *?lhs s = ?rhs s*)
proof (*rule analytic-continuation-open*[**where** $f = ?lhs$])
show *connected* $\{s.\ Re\ s > 0\}$

by (*intro convex-connected convex-halfspace-Re-gt*)
show {*s*. *Re s* > *1*} ≠ {} **by** (*auto intro*: *exI*[*of* - *2*])
show (*λs*. *rGamma* (*1* − *s*) ∗ *eval-fds* (*fds-perzeta a*) *s*) *holomorphic-on* {*s*. *0* < *Re s*}
unfolding *perzeta-def* **using** *a*
by (*auto intro*!: *holomorphic-intros le-less-trans*[*OF conv-abscissa-perzeta'*] *elim*!: *Ints-cases*)
show *?rhs holomorphic-on* {*s*. *0* < *Re s*} **using** *assms* **by** (*auto intro*!: *holomorphic-intros*)
next
fix *s* **assume** *s*: *s* ∈ {*s*. *Re s* > *1*}
have [*simp*]: *fds-perzeta* (*1* − *a*) = *fds-perzeta* (−*a*)
using *fds-perzeta.plus-of-nat*[*of* −*a 1*] **by** *simp*
have [*simp*]: *fds-perzeta* (*a* − *1*) = *fds-perzeta a*
using *fds-perzeta.minus-of-nat*[*of a 1*] **by** *simp*
from *s* **have** [*simp*]: *Gamma s* ≠ *0* **by** (*auto simp*: *Gamma-eq-zero-iff elim*!: *nonpos-Ints-cases*)

have (*2* ∗ *pi*) *powr* (−*s*) ∗ (i ∗ (i *powr* (−*s*) ∗ (*rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*)) −
i *powr s* ∗ (*rGamma s* ∗ *hurwitz-zeta* (*1* − *a*) (*1* − *s*)))) =
(*2* ∗ *pi*) *powr* (−*s*) ∗ ((i *powr* (*1* − *s*) ∗ (*rGamma s* ∗ *hurwitz-zeta a* (*1* − *s*)) +
i *powr* (*s* − *1*) ∗ (*rGamma s* ∗ *hurwitz-zeta* (*1* − *a*) (*1* − *s*))))
by (*simp add*: *powr-diff field-simps powr-minus*)
also have . . . = ((*2* ∗ *pi*) *powr* (−*s*)) ^ *2* ∗ (
eval-fds (*fds-perzeta a*) *s* ∗ (i *powr s* ∗ i *powr* (*s* − *1*) + i *powr* (−*s*) ∗ i *powr* (*1* − *s*)) +
eval-fds (*fds-perzeta* (−*a*)) *s* ∗ (i *powr s* ∗ i *powr* (*1* − *s*) + i *powr* (−*s*) ∗ i *powr* (*s* − *1*)))
using *s a* **by** (*subst* (*1 2*) *hurwitz-zeta-formula-aux*) (*auto simp*: *algebra-simps power2-eq-square*)
also have (i *powr s* ∗ i *powr* (*1* − *s*) + i *powr* (−*s*) ∗ i *powr* (*s* − *1*)) =
exp (i ∗ *complex-of-real pi* / *2*) + *exp* (− (i ∗ *complex-of-real pi* / *2*))
by (*simp add*: *powr-def exp-add* [*symmetric*] *field-simps*)
also have . . . = *0* **by** (*simp add*: *exp-eq-polar*)
also have i *powr s* ∗ i *powr* (*s* − *1*) = i *powr* (*2* ∗ *s* − *1*)
by (*simp add*: *powr-def exp-add* [*symmetric*] *field-simps*)
also have i *powr* (−*s*) ∗ i *powr* (*1* − *s*) = i *powr* (*1* − *2* ∗ *s*)
by (*simp add*: *powr-def exp-add* [*symmetric*] *field-simps*)
also have i *powr* (*2* ∗ *s* − *1*) + i *powr* (*1* − *2* ∗ *s*) = *2* ∗ *cos* ((*2* ∗ *s* − *1*) ∗ *pi* / *2*)
by (*simp add*: *powr-def cos-exp-eq algebra-simps minus-divide-left cos-sin-eq*)
also have . . . = *2* ∗ *sin* (*pi* − *s* ∗ *pi*) **by** (*simp add*: *cos-sin-eq field-simps*)
also have . . . = *2* ∗ *sin* (*s* ∗ *pi*) **by** (*simp add*: *sin-diff*)
finally have i ∗ (*rGamma s* ∗ i *powr* (−*s*) ∗ *hurwitz-zeta a* (*1* − *s*) −
rGamma s ∗ i *powr s* ∗ *hurwitz-zeta* (*1* − *a*) (*1* − *s*)) =
2 ∗ (*2* ∗ *pi*) *powr* −*s* ∗ *sin* (*s* ∗ *pi*) ∗ *eval-fds* (*fds-perzeta a*) *s*

by (*simp add*: *power2-eq-square mult-ac*)
hence (*2* ∗ *pi*) *powr s* / *2* ∗ i ∗
(i *powr* (−*s*) ∗ *hurwitz-zeta a* (*1* − *s*) −
i *powr s* ∗ *hurwitz-zeta* (*1* − *a*) (*1* − *s*)) =
Gamma s ∗ *sin* (*s* ∗ *pi*) ∗ *eval-fds* (*fds-perzeta a*) *s*
by (*subst* (*asm*) (*2*) *powr-minus*) (*simp add*: *field-simps rGamma-inverse-Gamma*)
also have *Gamma s* ∗ *sin* (*s* ∗ *pi*) = *pi* ∗ *rGamma* (*1* − *s*)
using *Gamma-reflection-complex*[*of s*]
by (*auto simp*: *divide-simps rGamma-inverse-Gamma mult-ac split*: *if-splits*)
finally show *?lhs s* = *?rhs s* **by** (*simp add*: *powr-diff*)
qed (*insert s*, *auto simp*: *open-halfspace-Re-gt*)

We can now use the above equation as a defining equation to continue the periodic zeta function F to the entire complex plane except at non-negative integer values for s. However, the positive integers are already covered by the original Dirichlet series definition of F, so we only need to take care of $s = 0$. We do this by cancelling the pole of Γ at 0 with the zero of $i^{-s}\zeta(1-s,a) - i^s\zeta(1-s,1-a)$.

lemma
assumes $q' \notin \mathbb{Z}$
shows *holomorphic-perzeta'*: *perzeta q' holomorphic-on A*
and *perzeta-altdef2*: *Re s* > *0* ⟹ *perzeta q' s* = *eval-fds* (*fds-perzeta q'*) *s*
proof −
define *q* **where** *q* = *frac q'*
from *assms* **have** *q*: *q* ∈ {*0*<..<*1*} **by** (*auto simp*: *q-def frac-lt-1*)
hence [*simp*]: *q* ∉ ℤ **by** (*auto elim*!: *Ints-cases*)
have [*simp*]: *frac q* = *q* **by** (*simp add*: *q-def frac-def*)
define *f* **where** *f* = (λ*s*. *complex-of-real* (*2* ∗ *pi*) *powr* (*s* − *1*) ∗ i ∗ *Gamma* (*1* − *s*) ∗
(i *powr* (−*s*) ∗ *hurwitz-zeta q* (*1* − *s*) −
i *powr s* ∗ *hurwitz-zeta* (*1* − *q*) (*1* − *s*)))

{
fix *s* :: *complex* **assume** $1 - s \in \mathbb{Z}_{\leq 0}$
then obtain *n* **where** *1* − *s* = *of-int n n* ≤ *0* **by** (*auto elim*!: *nonpos-Ints-cases*)
hence *s* = *1* − *of-int n* **by** (*simp add*: *algebra-simps*)
also have ... ∈ ℕ **using** ‹*n* ≤ *0*› **by** (*auto simp*: *Nats-altdef1 intro*: *exI*[*of* -*1* − *n*])
finally have *s* ∈ ℕ **.**
} **note** ∗ = *this*
hence *f holomorphic-on* −ℕ **using** *q*
by (*auto simp*: *f-def Nats-altdef2 nonpos-Ints-altdef not-le intro*!: *holomorphic-intros*)
also have *?this* ⟷ *perzeta q holomorphic-on* −ℕ **using** *assms*
by (*intro holomorphic-cong refl*) (*auto simp*: *perzeta-def Let-def f-def*)
finally have *holo*: *perzeta q holomorphic-on* −ℕ **.**

have *f-altdef*: *f s* = *eval-fds* (*fds-perzeta q*) *s* **if** *Re s* > *0* **and** *s* ∉ ℕ **for** *s*
using *perzeta-conv-hurwitz-zeta-aux*[*OF q*, *of s*] *that* ∗

by (*auto simp*: *rGamma-inverse-Gamma Gamma-eq-zero-iff divide-simps f-def perzeta-def*
split: *if-splits*)
show *perzeta q′ s* = *eval-fds* (*fds-perzeta q′*) *s* **if** *Re s* > *0* **for** *s*
using *f-altdef*[*of s*] *that assms* **by** (*auto simp*: *f-def perzeta-def Let-def q-def*)

have *cont*: *isCont* (*perzeta q*) *s* **if** *s* ∈ ℕ **for** *s*
proof (*cases s* = *0*)
case *False*
with *that* **obtain** *n* **where** [*simp*]: *s* = *of-nat n* **and** *n*: *n* > *0*
by (*auto elim*!: *Nats-cases*)
have ∗: *open* ({*s*. *Re s* > *0*} − (ℕ − {*of-nat n*})) **using** *Nats-subset-Ints*
by (*intro open-Diff closed-subset-Ints open-halfspace-Re-gt*) *auto*
have *eventually* (λ*s*. *s* ∈ {*s*. *Re s* > *0*} − (ℕ − {*of-nat n*})) (*nhds* (*of-nat n*)) **using** ‹*n* > *0*›
by (*intro eventually-nhds-in-open* ∗) *auto*
hence *ev*: *eventually* (λ*s*. *eval-fds* (*fds-perzeta q*) *s* = *perzeta q s*) (*nhds* (*of-nat n*))
proof *eventually-elim*
case (*elim s*)
thus *?case* **using** *q f-altdef*[*of s*]
by (*auto simp*: *perzeta-def dist-of-nat f-def elim*!: *Nats-cases Ints-cases*)
qed
have *isCont* (*eval-fds* (*fds-perzeta q*)) (*of-nat n*) **using** *q* **and** ‹*n* > *0*›
by (*intro continuous-eval-fds le-less-trans*[*OF conv-abscissa-perzeta′*])
(*auto elim*!: *Ints-cases*)
also have *?this* ⟷ *isCont* (*perzeta q*) (*of-nat n*) **using** *ev*
by (*intro isCont-cong ev*)
finally show *?thesis* **by** *simp*
next
assume [*simp*]: *s* = *0*
define *a* **where** *a* = *Complex* (*ln q*) (−*pi* / *2*)
define *b* **where** *b* = *Complex* (*ln* (*1* − *q*)) (*pi* / *2*)
have *eventually* (λ*s*::*complex*. *s* ∉ ℕ) (*at 0*)
unfolding *eventually-at-topological* **using** *Nats-subset-Ints*
by (*intro exI*[*of* - −(ℕ−{*0*})] *conjI open-Compl closed-subset-Ints*) *auto*
hence *ev*: *eventually* (λ*s*. *perzeta q s* = (*2* ∗ *pi*) *powr* (*s* − *1*) ∗ *Gamma* (*1* − *s*) ∗ i ∗
(i *powr* − *s* ∗ *pre-zeta q* (*1* − *s*) −i *powr s* ∗ *pre-zeta* (*1* − *q*) (*1* − *s*) +
(*exp* (*b* ∗ *s*) − *exp* (*a* ∗ *s*)) / *s*)) (*at* (*0*::*complex*))
(**is** *eventually* (λ*s*. - = *?f s*) -)
proof *eventually-elim*
case (*elim s*)
have *perzeta q s* = (*2* ∗ *pi*) *powr* (*s* − *1*) ∗ *Gamma* (*1* − *s*) ∗ i ∗
(i *powr* (−*s*) ∗ *hurwitz-zeta q* (*1* − *s*) −
i *powr s* ∗ *hurwitz-zeta* (*1* − *q*) (*1* − *s*)) (**is** - = - ∗ *?T*)
using *elim* **by** (*auto simp*: *perzeta-def powr-diff powr-minus field-simps*)
also have *?T* = i *powr* (−*s*) ∗ *pre-zeta q* (*1* − *s*) − i *powr s* ∗ *pre-zeta* (*1* −

q) (*1* − *s*) +
(i *powr s* ∗ (*1* − *q*) *powr s* − i *powr* (−*s*) ∗ *q powr s*) / *s* **using** *elim*
by (*auto simp*: *hurwitz-zeta-def field-simps*)
also have i *powr s* ∗ (*1* − *q*) *powr s* = *exp* (*b* ∗ *s*) **using** *q*
by (*simp add*: *powr-def exp-add algebra-simps Ln-Reals-eq Complex-eq b-def*)
also have i *powr* (−*s*) ∗ *q powr s* = *exp* (*a* ∗ *s*) **using** *q*
by (*simp add*: *powr-def exp-add Ln-Reals-eq exp-diff exp-minus diff-divide-distrib*

ring-distribs inverse-eq-divide mult-ac Complex-eq a-def)
finally show *?case* **.**
qed

have [*simp*]: ¬(*pi* ≤ *0*) **using** *pi-gt-zero* **by** (*simp add*: *not-le*)
have (λ*s*::*complex*. *if s* = *0 then b* − *a else* (*exp* (*b* ∗ *s*) − *exp* (*a* ∗ *s*)) / *s*)
has-fps-expansion (*fps-exp b* − *fps-exp a*) / *fps-X* (**is** *?f′ has-fps-expansion* -)
by (*rule fps-expansion-intros*)+ (*auto intro*!: *subdegree-geI simp*: *Ln-Reals-eq a-def b-def*)
hence *isCont ?f′ 0* **by** (*rule has-fps-expansion-imp-continuous*)
hence *?f′ −0→ b* − *a* **by** (*simp add*: *isCont-def*)
also have *?this* ⟷ (λ*s*. (*exp* (*b* ∗ *s*) − *exp* (*a* ∗ *s*)) / *s*) *−0→ b* − *a*
by (*intro filterlim-cong refl*) (*auto simp*: *eventually-at intro*: *exI*[*of* - *1*])
finally have *?f −0→ of-real* (*2* ∗ *pi*) *powr* (*0* − *1*) ∗ *Gamma* (*1* − *0*) ∗ i ∗
(i *powr −0* ∗ *pre-zeta q* (*1* − *0*) −i *powr 0* ∗ *pre-zeta* (*1* − *q*) (*1* − *0*) + (*b* − *a*))
(**is** *filterlim* - (*nhds ?c*) -)
using *q* **by** (*intro tendsto-intros isContD*)
(*auto simp*: *complex-nonpos-Reals-iff intro*!: *continuous-intros*)
also have *?c* = *perzeta q 0* **using** *q*
by (*simp add*: *powr-minus perzeta-def Ln-Reals-eq a-def b-def*
Complex-eq mult-ac inverse-eq-divide)
also have *?f −0→* . . . ⟷ *perzeta q −0→* . . .
by (*rule sym*, *intro filterlim-cong refl ev*)
finally show *isCont* (*perzeta q*) *s* **by** (*simp add*: *isCont-def*)
qed

have *perzeta q field-differentiable at s* **for** *s*
proof (*cases s* ∈ ℕ)
case *False*
with *holo* **have** *perzeta q field-differentiable at s within* −ℕ
unfolding *holomorphic-on-def* **by** *blast*
also have *at s within* −ℕ = *at s* **using** *False*
by (*intro at-within-open*) *auto*
finally show *?thesis* **.**
next
case *True*
hence ∗: *perzeta q holomorphic-on* (*ball s 1* − {*s*})
by (*intro holomorphic-on-subset*[*OF holo*]) (*auto elim*!: *Nats-cases simp*:

dist-of-nat)
  have *perzeta q holomorphic-on ball s 1* **using** *cont True*
   by (*intro no-isolated-singularity'*[*OF - ∗*])
     (*auto simp*: *at-within-open*[*of - ball s 1*] *isCont-def*)
  hence *perzeta q field-differentiable at s within ball s 1*
   unfolding *holomorphic-on-def* **by** *auto*
  thus *?thesis* **by** (*simp add*: *at-within-open*[*of - ball s 1*])
 qed
 hence *perzeta q holomorphic-on UNIV*
  by (*auto simp*: *holomorphic-on-def*)
 also have *perzeta q = perzeta q'* **by** (*simp add*: *q-def*)
 finally show *perzeta q' holomorphic-on A* **by** *auto*
qed

lemma *perzeta-altdef1*: *Re s > 1* $\Longrightarrow$ *perzeta q' s = eval-fds* (*fds-perzeta q'*) *s*
 by (*cases q'* $\in \mathbb{Z}$) (*auto simp*: *perzeta-int eval-fds-zeta fds-perzeta-int perzeta-altdef2*)

lemma *holomorphic-perzeta*: *q* $\notin \mathbb{Z} \vee$ *1* $\notin$ *A* $\Longrightarrow$ *perzeta q holomorphic-on A*
 by (*cases q* $\in \mathbb{Z}$) (*auto simp*: *perzeta-int intro*: *holomorphic-perzeta' holomorphic-zeta*)

lemma *holomorphic-perzeta''* [*holomorphic-intros*]:
 assumes *f holomorphic-on A* **and** *q* $\notin \mathbb{Z} \vee$ ($\forall$ *x*$\in$*A*. *f x* $\neq$ *1*)
 shows (λ*x*. *perzeta q* (*f x*)) *holomorphic-on A*
proof −
 have *perzeta q* $\circ$ *f holomorphic-on A* **using** *assms*
  by (*intro holomorphic-on-compose holomorphic-perzeta*) *auto*
 thus *?thesis* **by** (*simp add*: *o-def*)
qed

Using this analytic continuation of the periodic zeta function, Hurwitz's formula now holds (almost) on the entire complex plane.

theorem *hurwitz-zeta-formula*:
 fixes *a* :: *real* **and** *s* :: *complex*
 assumes *a* $\in$ {*0<..1*} **and** *s* $\neq$ *0* **and** *a* $\neq$ *1* $\vee$ *s* $\neq$ *1*
 shows *rGamma s ∗ hurwitz-zeta a* (*1* − *s*) =
   (*2 ∗ pi*) *powr* − *s ∗* (i *powr* − *s ∗ perzeta a s* + i *powr s ∗ perzeta* (−*a*) *s*)
 (**is** *?f s = ?g s*)
proof −
 define *A* **where** *A = UNIV* − (*if a* $\in \mathbb{Z}$ *then* {*0*, *1*} *else* {*0* :: *complex*})
 show *?thesis*
 proof (*rule analytic-continuation-open*[**where** *f = ?f*])
  show *?f holomorphic-on A* **using** *assms* **by** (*auto intro*!: *holomorphic-intros simp*: *A-def*)
  show *?g holomorphic-on A* **using** *assms*
   by (*auto intro*!: *holomorphic-intros simp*: *A-def minus-in-Ints-iff*)
 next
  fix *s* **assume** *s* $\in$ {*s*. *Re s > 1*}

thus *?f s = ?g s* **using** *hurwitz-zeta-formula-aux[of a s] assms*
by (*simp add*: *perzeta-altdef1*)
qed (*insert assms*, *auto simp*: *open-halfspace-Re-gt A-def elim*!: *Ints-cases*
intro: *connected-open-delete-finite exI[of - 2]*)
qed

The equation expressing the periodic zeta function in terms of the Hurwitz zeta function can be extened similarly.

theorem *perzeta-conv-hurwitz-zeta*:
fixes *a* :: *real* **and** *s* :: *complex*
assumes $a \in \{0{<}..{<}1\}$ **and** $s \neq 0$
shows *rGamma* (*1* − *s*) ∗ *perzeta a s* =
(*2* ∗ *pi*) *powr* (*s* − *1*) ∗ i ∗ (i *powr* (−*s*) ∗ *hurwitz-zeta a* (*1* − *s*) −
i *powr s* ∗ *hurwitz-zeta* (*1* − *a*) (*1* − *s*))
(**is** *?f s = ?g s*)
proof (*rule analytic-continuation-open*[**where** *f = ?f*])
show *?f holomorphic-on* −{*0*} **using** *assms* **by** (*auto intro*!: *holomorphic-intros*
elim: *Ints-cases*)
show *?g holomorphic-on* −{*0*} **using** *assms* **by** (*auto intro*!: *holomorphic-intros*)
next
fix *s* **assume** *s* ∈ {*s*. *Re s* > *1*}
thus *?f s = ?g s* **using** *perzeta-conv-hurwitz-zeta-aux[of a s] assms*
by (*simp add*: *perzeta-altdef1*)
qed (*insert assms*, *auto simp*: *open-halfspace-Re-gt connected-punctured-universe*
intro: *exI[of - 2]*)

As a simple corollary, we derive the reflection formula for the Riemann zeta function:

corollary *zeta-reflect*:
fixes *s* :: *complex*
assumes *s* ≠ *0 s* ≠ *1*
shows *rGamma s* ∗ *zeta* (*1* − *s*) = *2* ∗ (*2* ∗ *pi*) *powr* −*s* ∗ *cos* (*s* ∗ *pi* / *2*) ∗
zeta s
using *hurwitz-zeta-formula[of 1 s] assms*
by (*simp add*: *zeta-def cos-exp-eq powr-def perzeta-int algebra-simps*)

corollary *zeta-reflect'*:
fixes *s* :: *complex*
assumes *s* ≠ *0 s* ≠ *1*
shows *rGamma* (*1* − *s*) ∗ *zeta s* = *2* ∗ (*2* ∗ *pi*) *powr* (*s* − *1*) ∗ *sin* (*s* ∗ *pi* /
2) ∗ *zeta* (*1* − *s*)
using *zeta-reflect[of 1 − s] assms* **by** (*simp add*: *cos-sin-eq field-simps*)

It is now easy to see that all the non-trivial zeroes of the Riemann zeta function must lie the critical strip $(0; 1)$, and they must be symmetric around the $\mathfrak{R}(z) = \frac{1}{2}$ line.

corollary *zeta-zeroD*:
assumes *zeta s* = *0 s* ≠ *1*

```
  shows   Re s ∈ {0<..<1} ∨ (∃ n::nat. n > 0 ∧ even n ∧ s = −real n)
proof (cases Re s ≤ 0)
  case False
  with zeta-Re-ge-1-nonzero[of s] assms have Re s < 1
    by (cases Re s < 1) auto
  with False show ?thesis by simp
next
  case True
  {
    assume ∗: ⋀n. n > 0 ⟹ even n ⟹ s ≠ −real n
    have s ≠ of-int n for n :: int
    proof
      assume [simp]: s = of-int n
      show False
      proof (cases n 0::int rule: linorder-cases)
        assume n < 0
        show False
        proof (cases even n)
          case True
          hence nat (−n) > 0 even (nat (−n)) using ‹n < 0›
            by (auto simp: even-nat-iff)
          with ∗ have s ≠ −real (nat (−n)) .
          with ‹n < 0› and True show False by auto
        next
          case False
          with ‹n < 0› have of-int n = (−of-nat (nat (−n)) :: complex) by simp
           also have zeta ... = −(bernoulli′ (Suc (nat (−n)))) / of-nat (Suc (nat
(−n)))
            using ‹n < 0› by (subst zeta-neg-of-nat) (auto)
          finally have bernoulli′ (Suc (nat (−n))) = 0 using assms
            by (auto simp del: of-nat-Suc)
          with False and ‹n < 0› show False
            by (auto simp: bernoulli′-zero-iff even-nat-iff)
        qed
      qed (insert assms True, auto)
    qed
    hence rGamma s ≠ 0
      by (auto simp: rGamma-eq-zero-iff nonpos-Ints-def)
    moreover from assms have [simp]: s ≠ 0 by auto
    ultimately have zeta (1 − s) = 0 using zeta-reflect[of s] and assms
      by auto
    with True zeta-Re-ge-1-nonzero[of 1 − s] have Re s > 0 by auto
  }
  with True show ?thesis by auto
qed

lemma zeta-zero-reflect:
  assumes Re s ∈ {0<..<1} and zeta s = 0
  shows   zeta (1 − s) = 0
```

proof −
  from *assms* **have** *rGamma s* ≠ *0*
    by (*auto simp*: *rGamma-eq-zero-iff elim*!: *nonpos-Ints-cases*)
  moreover from *assms* **have** *s* ≠ *0* **and** *s* ≠ *1* **by** *auto*
  ultimately show *?thesis* **using** *zeta-reflect*[*of s*] **and** *assms* **by** *auto*
qed

corollary *zeta-zero-reflect-iff*:
  assumes *Re s* ∈ {*0*<..<*1*}
  shows *zeta* (*1* − *s*) = *0* ⟷ *zeta s* = *0*
  using *zeta-zero-reflect*[*of s*] *zeta-zero-reflect*[*of 1* − *s*] *assms* **by** *auto*

2.10 More functional equations

lemma *perzeta-conv-hurwitz-zeta-multiplication*:
  fixes *k* :: *nat* **and** *a* :: *int* **and** *s* :: *complex*
  assumes *k* > *0 s* ≠ *1*
  shows *k powr s* ∗ *perzeta* (*a* / *k*) *s* =
      (∑ *n*=*1*..*k*. *exp* (*2* ∗ *pi* ∗ *n* ∗ *a* / *k* ∗ i) ∗ *hurwitz-zeta* (*n* / *k*) *s*)
  (**is** *?lhs s* = *?rhs s*)
proof (*rule analytic-continuation-open*[**where** *?f* = *?lhs* **and** *?g* = *?rhs*])
  show *connected* (−{*1*::*complex*}) **by** (*rule connected-punctured-universe*) *auto*
  show {*s*. *Re s* > *1*} ≠ {} **by** (*auto intro*!: *exI*[*of* - *2*])
next
  fix *s* **assume** *s*: *s* ∈ {*s*. *Re s* > *1*}
  let *?f* = λ*n*. *exp* (*2* ∗ *pi* ∗ *n* ∗ *a* / *k* ∗ i)

  show *?lhs s* = *?rhs s*
  proof (*rule sums-unique2*)
    have (λ*m*. ∑ *n*=*1*..*k*. *?f n* ∗ (*of-nat m* + *of-real* (*real n* / *real k*)) *powr* −*s*)
sums
        (∑ *n*=*1*..*k*. *?f n* ∗ *hurwitz-zeta* (*real n* / *real k*) *s*)
      using *assms s* **by** (*intro sums-sum sums-mult sums-hurwitz-zeta*) *auto*
    also have (λ*m*. ∑ *n*=*1*..*k*. *?f n* ∗ (*of-nat m* + *of-real* (*real n* / *real k*)) *powr*
−*s*) =
          (λ*m*. *of-nat k powr s* ∗ (∑ *n*=*1*..*k*. *?f n* ∗ *of-nat* (*m* ∗ *k* + *n*) *powr*
−*s*))
      unfolding *sum-distrib-left*
    proof (*intro ext sum.cong*, *goal-cases*)
      case (*2 m n*)
      hence *m* ∗ *k* + *n* > *0* **by** (*intro add-nonneg-pos*) *auto*
     hence *of-nat 0* ≠ (*of-nat* (*m* ∗ *k* + *n*) :: *complex*) **by** (*simp only*: *of-nat-eq-iff*)
      also have *of-nat* (*m* ∗ *k* + *n*) = *of-nat m* ∗ *of-nat k* + (*of-nat n* :: *complex*)
by *simp*
      finally have *nz*: . . . ≠ *0* **by** *auto*

      have *of-nat m* + *of-real* (*real n* / *real k*) =
          (*inverse* (*of-nat k*) ∗ *of-nat* (*m* ∗ *k* + *n*) :: *complex*) **using** *assms*

by (*simp add*: *field-simps del*: *div-mult-self1 div-mult-self2 div-mult-self3 div-mult-self4*)
also from *nz* **have** ... *powr* −*s* = *of-nat k powr s* ∗ *of-nat* (*m* ∗ *k* + *n*) *powr* −*s*
by (*subst powr-times-real*) (*auto simp*: *add-eq-0-iff powr-def exp-minus Ln-inverse*)
finally show *?case* **by** *simp*
qed *auto*
finally show ... *sums* (∑ *n*=*1*..*k*. *?f n* ∗ *hurwitz-zeta* (*real n* / *real k*) *s*) **.**
next
define *g* **where** *g* = (λ*m*. *exp* (*2* ∗ *pi* ∗ i ∗ *m* ∗ (*real-of-int a* / *real k*)))
have (λ*m*. *g* (*Suc m*) / (*Suc m*) *powr s*) *sums eval-fds* (*fds-perzeta* (*a* / *k*)) *s*
unfolding *g-def* **using** *s* **by** (*intro sums-fds-perzeta*) *auto*
also have (λ*m*. *g* (*Suc m*) / (*Suc m*) *powr s*) = (λ*m*. *?f* (*Suc m*) ∗ (*Suc m*) *powr* −*s*)
by (*simp add*: *powr-minus field-simps g-def*)
also have *eval-fds* (*fds-perzeta* (*a* / *k*)) *s* = *perzeta* (*a* / *k*) *s*
using *s* **by** (*simp add*: *perzeta-altdef1*)
finally have (λ*m*. ∑ *n*=*m*∗*k*..<*m*∗*k*+*k*. *?f* (*Suc n*) ∗ *of-nat* (*Suc n*) *powr* −*s*) *sums perzeta* (*a* / *k*) *s*
using ‹*k* > *0*› **by** (*rule sums-group*)
also have (λ*m*. ∑ *n*=*m*∗*k*..<*m*∗*k*+*k*. *?f* (*Suc n*) ∗ *of-nat* (*Suc n*) *powr* −*s*) =
(λ*m*. ∑ *n*=*1*..*k*. *?f* (*m* ∗ *k* + *n*) ∗ *of-nat* (*m* ∗ *k* + *n*) *powr* −*s*)
proof (*rule ext*, *goal-cases*)
case (*1 m*)
show *?case* **using** *assms*
by (*intro ext sum.reindex-bij-witness*[*of* - λ*n*. *m* ∗ *k* + *n* − *1* λ*n*. *Suc n* − *m* ∗ *k*]) *auto*
qed
also have (λ*m n*. *?f* (*m* ∗ *k* + *n*)) = (λ*m n*. *?f n*)
proof (*intro ext*)
fix *m n* :: *nat*
have *?f* (*m* ∗ *k* + *n*) / *?f n* = *exp* (*2* ∗ *pi* ∗ *m* ∗ *a* ∗ i)
using ‹*k* > *0*› **by** (*auto simp*: *ring-distribs add-divide-distrib exp-add mult-ac*)
also have ... = *cis* (*2* ∗ *pi* ∗ (*m* ∗ *a*))
by (*simp add*: *exp-eq-polar mult-ac*)
also have ... = *1*
by (*rule cis-multiple-2pi*) *auto*
finally show *?f* (*m* ∗ *k* + *n*) = *?f n*
by *simp*
qed
finally show (λ*m*. *of-nat k powr s* ∗ (∑ *n*=*1*..*k*. *?f n* ∗ *of-nat* (*m* ∗ *k* + *n*) *powr* −*s*)) *sums*
(*of-nat k powr s* ∗ *perzeta* (*a* / *k*) *s*) **by** (*rule sums-mult*)
qed
qed (*use assms* **in** ‹*auto intro*!: *holomorphic-intros simp*: *finite-imp-closed open-halfspace-Re-gt*›)

lemma *perzeta-conv-hurwitz-zeta-multiplication*′:
fixes *k* :: *nat* **and** *a* :: *int* **and** *s* :: *complex*

assumes $k > 0$ $s \neq 1$
shows *perzeta* $(a\ /\ k)$ $s = k$ *powr* $-s *$
$(\sum n{=}1..k.\ exp\ (2 * pi * n * a\ /\ k * \mathrm{i}) * hurwitz\text{-}zeta\ (n\ /\ k)\ s)$
using *perzeta-conv-hurwitz-zeta-multiplication*[*of k s a*] *assms*
by (*simp add*: *powr-minus field-simps*)

lemma *zeta-conv-hurwitz-zeta-multiplication*:
fixes *k a* :: *nat* **and** *s* :: *complex*
assumes $k > 0$ $s \neq 1$
shows k *powr* $s * zeta\ s = (\sum n{=}1..k.\ hurwitz\text{-}zeta\ (n\ /\ k)\ s)$
using *perzeta-conv-hurwitz-zeta-multiplication*[*of k s 0*]
using *assms* **by** (*simp add*: *perzeta-int*)

lemma *hurwitz-zeta-one-half-left*:
assumes $s \neq 1$
shows *hurwitz-zeta* $(1\ /\ 2)$ $s = (2$ *powr* $s - 1) * zeta\ s$
using *zeta-conv-hurwitz-zeta-multiplication*[*of 2 s*] *assms*
by (*simp add*: *eval-nat-numeral zeta-def field-simps*)

theorem *hurwitz-zeta-functional-equation*:
fixes *h k* :: *nat* **and** *s* :: *complex*
assumes *hk*: $k > 0$ $h \in \{0{<}..k\}$ **and** *s*: $s \notin \{0,\ 1\}$
defines $a \equiv$ *real h / real k*
shows *rGamma* $s *$ *hurwitz-zeta* $a\ (1 - s) =$
$2 * (2 * pi * k)$ *powr* $-s *$
$(\sum n{=}1..k.\ cos\ (s{*}pi/2 - 2{*}pi{*}n{*}h/k) * hurwitz\text{-}zeta\ (n\ /\ k)\ s)$
proof −
from *hk* **have** *a*: $a \in \{0{<}..1\}$ **by** (*auto simp*: *a-def*)

have *rGamma* $s *$ *hurwitz-zeta* $a\ (1 - s) =$
$(2 * pi)$ *powr* $-\ s * (\mathrm{i}$ *powr* $-\ s *$ *perzeta* $a\ s + \mathrm{i}$ *powr* $s *$ *perzeta* $(-\ a)\ s)$
using *s a* **by** (*intro hurwitz-zeta-formula*) *auto*
also have $\ldots = (2 * pi)$ *powr* $-\ s * (\mathrm{i}$ *powr* $-\ s *$ *perzeta* (*of-int* (*int h*) / *k*) *s*
+
i *powr* $s *$ *perzeta* (*of-int* (−*int h*) / *k*) *s*)
by (*simp add*: *a-def*)
also have $\ldots = (2 * pi)$ *powr* $-s * k$ *powr* $-s *$
$((\sum n{=}1..k.\ \mathrm{i}\ powr\ -s * cis\ (2 * pi * n * h\ /\ k) * hurwitz\text{-}zeta\ (n\ /\ k)\ s) +$
$(\sum n{=}1..k.\ \mathrm{i}\ powr\ s\ * cis\ (-2 * pi * n * h\ /\ k) * hurwitz\text{-}zeta\ (n\ /\ k)\ s))$
(**is** - = - * (*?S1* + *?S2*)) **using** *hk a s*
by (*subst* (*1 2*) *perzeta-conv-hurwitz-zeta-multiplication'*)
(*auto simp*: *field-simps sum-distrib-left sum-distrib-right exp-eq-polar*)
also have $(2 * pi)$ *powr* $-s * k$ *powr* $-s = (2 * k * pi)$ *powr* $-s$
using *hk pi-gt-zero*
by (*simp add*: *powr-def Ln-times-Reals field-simps exp-add exp-diff exp-minus*)
also have $?S1 + ?S2 = (\sum n{=}1..k.\ (\mathrm{i}\ powr\ -s * cis\ (2{*}pi{*}n{*}h/k) + \mathrm{i}\ powr\ s$
$* cis\ (-2{*}pi{*}n{*}h/k)) *$
hurwitz-zeta $(n\ /\ k)\ s)$
(**is** - = $(\sum n{\in}$-. *?c n* ∗ -)) **by** (*simp add*: *algebra-simps sum.distrib*)

also have *?c* = (*λn. 2* ∗ *cos* (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*))
proof
fix *n* :: *nat*
have i *powr* −*s* ∗ *cis* (*2*∗*pi*∗*n*∗*h*/*k*) = *exp* (−*s*∗*pi*/*2*∗i + *2*∗*pi*∗*n*∗*h*/*k*∗i)
unfolding *exp-add* **by** (*simp add*: *powr-def cis-conv-exp mult-ac*)
moreover have i *powr s* ∗ *cis* (−*2*∗*pi*∗*n*∗*h*/*k*) = *exp* (*s*∗*pi*/*2*∗i + −*2*∗*pi*∗*n*∗*h*/*k*∗i)
unfolding *exp-add* **by** (*simp add*: *powr-def cis-conv-exp mult-ac*)
ultimately have *?c n* = *exp* (i ∗ (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*)) + *exp* (−(i ∗ (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*)))
by (*simp add*: *mult-ac ring-distribs*)
also have ... / *2* = *cos* (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*)
by (*rule cos-exp-eq* [*symmetric*])
finally show *?c n* = *2* ∗ *cos* (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*)
by *simp*
qed
also have (*2* ∗ *k* ∗ *pi*) *powr* −*s* ∗ (∑ *n*=*1*..*k*. ... *n* ∗ *hurwitz-zeta* (*n* / *k*) *s*) =
2 ∗ (*2* ∗ *pi* ∗ *k*) *powr* −*s* ∗
(∑ *n*=*1*..*k*. *cos* (*s*∗*pi*/*2* − *2*∗*pi*∗*n*∗*h*/*k*) ∗ *hurwitz-zeta* (*n* / *k*) *s*)
by (*simp add*: *sum-distrib-left sum-distrib-right mult-ac*)
finally show *?thesis* **.**
qed

lemma *perzeta-one-half-left*: *s* ≠ *1* ⟹ *perzeta* (*1* / *2*) *s* = (*2 powr* (*1*−*s*) − *1*) ∗ *zeta s*
using *perzeta-conv-hurwitz-zeta-multiplication*′[*of 2 s 1*]
by (*simp add*: *eval-nat-numeral hurwitz-zeta-one-half-left powr-minus field-simps zeta-def powr-diff*)

lemma *perzeta-one-half-left*′:
perzeta (*1* / *2*) *s* =
(*if s* = *1 then* −*ln 2 else* (*2 powr* (*1* − *s*) − *1*) / (*s* − *1*)) ∗ ((*s* − *1*) ∗ *pre-zeta 1 s* + *1*)
by (*cases s* = *1*) (*auto simp*: *perzeta-one-half-left field-simps zeta-def hurwitz-zeta-def*)

end

3 The Laurent series expansion of ζ at 1

theory *Zeta-Laurent-Expansion*
imports *Zeta-Function*
begin

In this section, we shall derive the Laurent series expansion of $\zeta(s)$ at $s = 1$, which is of the form

$$\zeta(s) = \frac{1}{s-1} + \sum_{n=0}^{\infty} \frac{(-1)^n\,\gamma_n}{n!}(s-1)^n$$

where the γ_n are the *Stieltjes constants*. Notably, γ_0 is equal to the Euler–

Mascheroni constant γ.

3.1 Definition of the Stieltjes constants

We define the Stieltjes constants by their infinite series form, since it is fairly easy to show the convergence of the series by the comparison test.

definition *stieltjes-gamma* :: *nat* ⇒ *'a* :: *real-algebra-1* **where**
stieltjes-gamma n =
of-real ($\sum$*k. ln* (*k+1*) ^ *n* / (*k+1*) − (*ln* (*k+2*) ^(*n+1*) − *ln* (*k+1*) ^(*n* + *1*)) / (*n* + *1*))

lemma *stieltjes-gamma-0* [*simp*]: *stieltjes-gamma 0* = *euler-mascheroni*
using *euler-mascheroni-sum-real* **by** (*simp add*: *sums-iff stieltjes-gamma-def field-simps*)

lemma *stieltjes-gamma-summable*:
summable (λ*k. ln* (*k+1*) ^ *n* / (*k+1*) − (*ln* (*k+2*) ^(*n+1*) − *ln* (*k+1*) ^(*n* + *1*)) / (*n* + *1*))
(**is** *summable ?f*)
proof (*rule summable-comparison-test-bigo*)

have *eventually* (λ*x*::*real. ln x* ^ *n* − *ln x* ^(*n+1*) ∗ (*inverse* (*ln x*) ∗ (*1* + *real n*)) ∗
inverse (*real n* + *1*) = *0*) *at-top*
using *eventually-gt-at-top*[*of 1*] **by** *eventually-elim* (*auto simp*: *field-simps*)
thus *?f* ∈ *O*(λ*k. k powr* (−*3*/*2*))
by *real-asymp*
qed (*simp-all add*: *summable-real-powr-iff*)

lemma *of-real-stieltjes-gamma* [*simp*]: *of-real* (*stieltjes-gamma k*) = *stieltjes-gamma k*
by (*simp add*: *stieltjes-gamma-def*)

lemma *sums-stieltjes-gamma*:
(λ*k. ln* (*k+1*) ^ *n* / (*k+1*) − (*ln* (*k+2*) ^(*n+1*) − *ln* (*k+1*) ^(*n* + *1*)) / (*n* + *1*))
sums stieltjes-gamma n
using *stieltjes-gamma-summable*[*of n*] **unfolding** *stieltjes-gamma-def* **by** (*simp add*: *summable-sums*)

We can now derive the alternative definition of the Stieltjes constants as a limit. This limit can also be written in the Euler–MacLaurin-style form

$$\lim_{m\to\infty}\left(\sum_{k=1}^{m}\frac{\ln^n k}{k}-\int_1^m\frac{\ln^n x}{x}\,\mathrm{d}x\right),$$

which is perhaps a bit more illuminating.

lemma *stieltjes-gamma-real-limit-form*:

($\lambda m.$ ($\sum k = 1..m.$ *ln* (*real* k) ^ n / *real* k) − *ln* (*real* m) ^($n + 1$) / *real* ($n + 1$))
$\longrightarrow$ *stieltjes-gamma n*
proof −
have (λm::*nat*. $\sum k<m.$ *ln* (k+1) ^ n / (k+1) − (*ln* (k+2) ^(n+1) − *ln* (k+1) ^($n + 1$)) / ($n + 1$))
$\longrightarrow$ *stieltjes-gamma n*
using *sums-stieltjes-gamma*[*of n*] **by** (*simp add*: *add-ac sums-def*)
also have (λm::*nat*. $\sum k<m.$ *ln* (k+1) ^ n / (k+1) − (*ln* (k+2) ^(n+1) − *ln* (k+1) ^($n + 1$)) / ($n + 1$)) =
(λm::*nat*. ($\sum k$=1..m. *ln* k ^ n / k) − *ln* ($m + 1$) ^($n + 1$) / ($n + 1$))
(**is** *?lhs* = *?rhs*)
proof (*rule ext*, *goal-cases*)
fix m :: *nat*
have ($\sum k<m.$ *ln* (k+1) ^ n / (k+1) − (*ln* (k+2) ^(n+1) − *ln* (k+1) ^($n + 1$)) / ($n + 1$)) =
($\sum k<m.$ *ln* (k+1) ^ n / (k+1)) −
($\sum k<m.$ *ln* (*Suc k*+1) ^(n+1) − *ln* (k+1) ^($n + 1$)) / ($n + 1$)
by (*simp add*: *sum-subtractf flip*: *sum-divide-distrib*)
also have ($\sum k<m.$ *ln* (k+1) ^ n / (k+1)) = ($\sum k$=1..m. *ln* k ^ n / k)
by (*rule sum.reindex-bij-witness*[*of* - $\lambda k.$ k−1 *Suc*]) *auto*
also have ($\sum k<m.$ *ln* (*Suc k*+1) ^(n+1) − *ln* (k+1) ^($n + 1$)) = *ln* ($m + 1$) ^($n + 1$)
by (*subst sum-lessThan-telescope*) *simp-all*
finally show *?lhs m* = *?rhs m* **.**
qed
finally have ∗: ($\lambda m.$ ($\sum k = 1..m.$ *ln* k ^ n / k) − *ln* ($m + 1$) ^($n + 1$) / ($n + 1$))
$\longrightarrow$ *stieltjes-gamma n* **.**
have ∗∗: ($\lambda m.$ *ln* ($m + 1$) ^($n + 1$) / ($n + 1$) − *ln* m ^($n + 1$) / ($n + 1$)) $\longrightarrow$ *0*
by *real-asymp*
from *tendsto-add*[*OF* ∗ ∗∗] **show** *?thesis* **by** (*simp add*: *algebra-simps*)
qed

lemma *stieltjes-gamma-limit-form*:
($\lambda m.$ *of-real* (($\sum k$=1..m. *ln* (*real* k) ^ n / *real* k) − *ln* (*real* m) ^($n + 1$) / *real* ($n + 1$)))
$\longrightarrow$ (*stieltjes-gamma n* :: ′*a* :: *real-normed-algebra-1*)
proof −
have ($\lambda m.$ *of-real* (($\sum k$=1..m. *ln* (*real* k) ^ n / *real* k) − *ln* m ^($n + 1$) / *real* ($n + 1$)))
$\longrightarrow$ (*of-real* (*stieltjes-gamma n*) :: ′*a*)
using *stieltjes-gamma-real-limit-form*[*of n*] **by** (*intro tendsto-of-real*) (*auto simp*: *add-ac*)
thus *?thesis* **by** *simp*
qed

lemma *stieltjes-gamma-real-altdef*:

(*stieltjes-gamma n* :: *real*) =
lim (λm. ($\sum k$ = *1..m*. *ln* (*real k*) ^ *n* / *real k*) −
ln (*real m*) ^ (*n* + *1*) / *real* (*n* + *1*))
by (*rule sym*, *rule limI*, *rule stieltjes-gamma-real-limit-form*)

3.2 Proof of the Laurent expansion

We shall follow the proof by Briggs and Chowla [2], which examines the entire function $g(s) = (2^{1-s} - 1)\zeta(s)$. They determine the value of $g^{(k)}(1)$ in two different ways: First by the Dirichlet series of g and then by its power series expansion around 1. We shall do the same here.

context
fixes *g* **and** *G1 G2 G2′ G* :: *complex fps* **and** *A* :: *nat* ⇒ *complex*
defines *g* ≡ *perzeta* (*1* / *2*)
defines *G1* ≡ *fps-shift 1* (*fps-exp* (−*ln 2* :: *complex*) − *1*)
defines *G2* ≡ *fps-expansion* (λs. (*s* − *1*) ∗ *pre-zeta 1 s* + *1*) *1*
defines *G2′* ≡ *fps-expansion* (*pre-zeta 1*) *1*
defines *G* ≡ *G1* ∗ *G2*
defines *A* ≡ *fps-nth G2*
begin

G1, *G2*, *G2′*, and *G2* are the formal power series expansions of functions around *s* = *1* of the entire functions

- $(2^{1-s} - 1)/(s - 1)$,
- $(s - 1)\zeta(s)$,
- $\zeta(s) - \frac{1}{s-1}$,
- $(2^{1-s} - 1)\zeta(s)$,

respectively.

Our goal is to determine the coefficients of *G2′*, and we shall do so by determining the coefficients of *G2* (which are the same, but shifted by 1). This in turn will be done by determining the coefficients of *G* = *G1* ∗ *G2*.

Note that $(2^{1-s} - 1)\zeta(s)$ is written as *perzeta* (*1* / *2*) in Isabelle (using the periodic ζ function) and the analytic continuation of $\zeta(s) - \frac{1}{s-1}$ is written as *pre-zeta 1 s* (*pre-zeta* is an artefact from the definition of *zeta*, which comes in useful here).

lemma *stieltjes-gamma-aux1*: (λn. (−*1*)^(*n*+*1*) ∗ *ln*(*n*+*1*)^*k* / (*n*+*1*)) *sums* ((−*1*)^*k* ∗ (*deriv*^^*k*) *g 1*)
proof −
define *H* **where** *H* = *fds-perzeta* (*1* / *2*)
have *conv*: *conv-abscissa H* < *1* **unfolding** *H-def*
by (*rule le-less-trans*[*OF conv-abscissa-perzeta′*]) (*use fraction-not-in-ints*[*of 2 1*] **in** *auto*)

have [*simp*]: *eval-fds H s* = *g s* **if** *Re s* > *0* **for** *s*
unfolding *H-def g-def* **using** *fraction-not-in-ints*[*of 2 1*] *that*
by (*subst perzeta-altdef2*) *auto*
have *ev*: *eventually* (*λs. s* ∈ {*s. Re s* > *0*}) (*nhds 1*)
by (*intro eventually-nhds-in-open open-halfspace-Re-gt*) *auto*
have [*simp*]: (*deriv* ^^ *k*) (*eval-fds H*) *1* = (*deriv* ^^ *k*) *g 1*
by (*intro higher-deriv-cong-ev eventually-mono*[*OF ev*]) *auto*

have *fds-converges* ((*fds-deriv* ^^ *k*) *H*) *1*
by (*intro fds-converges le-less-trans*[*OF conv-abscissa-higher-deriv-le*])
(*use conv* **in** ‹*simp add*: *one-ereal-def*›)
hence (*λn. fds-nth* ((*fds-deriv* ^^ *k*) *H*) (*n+1*) / *real* (*n+1*)) *sums eval-fds* ((*fds-deriv* ^^ *k*) *H*) *1*
by (*simp add*: *fds-converges-altdef*)
also have *eval-fds* ((*fds-deriv* ^^ *k*) *H*) *1* = (*deriv* ^^ *k*) (*eval-fds H*) *1*
using *conv* **by** (*intro eval-fds-higher-deriv*) (*auto simp*: *one-ereal-def*)
also have (*λn. fds-nth* ((*fds-deriv* ^^ *k*) *H*) (*n+1*) / *real* (*n+1*)) =
(*λn.* (*−1*) ^ *k* ∗ (*−1*) ^ (*n+1*) ∗ *ln* (*real* (*n+1*)) ^ *k* / (*n+1*))
by (*auto simp*: *fds-nth-higher-deriv algebra-simps H-def fds-perzeta-one-half Ln-Reals-eq*)
finally have (*λn.* (− *1*) ^ *k* ∗ *complex-of-real* ((*−1*) ^ (*n+1*) ∗ *ln* (*real* (*n+1*)) ^ *k* / *real* (*n+1*))) *sums*
((*deriv* ^^ *k*) *g 1*) **by** (*simp add*: *algebra-simps*)

hence (*λn.* (*−1*) ^ *k* ∗ ((*−1*) ^ *k* ∗ *complex-of-real* ((*−1*) ^ (*n+1*) ∗ *ln* (*real* (*n+1*)) ^ *k* / *real* (*n+1*)))) *sums*
((*−1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*) **by** (*intro sums-mult*)
also have (*λn.* (*−1*) ^ *k* ∗ ((*−1*) ^ *k* ∗ *complex-of-real* ((*−1*) ^ (*n+1*) ∗ *ln* (*real* (*n+1*)) ^ *k* / *real* (*n+1*)))) =
(*λn. complex-of-real* ((*−1*) ^ (*n+1*) ∗ *ln* (*real* (*n+1*)) ^ *k* / *real* (*n+1*)))
by (*intro ext*) *auto*
finally show *?thesis* **.**
qed

lemma *stieltjes-gamma-aux2*: (*deriv* ^^ *k*) *g 1* = *fact k* ∗ *fps-nth G k*
and *stieltjes-gamma-aux3*: *G2* = *fps-X* ∗ *G2′* + *1*
proof −
have [*simp*]: *fps-conv-radius G1* = ∞
using *fps-conv-radius-diff*[*of fps-exp* (*−Ln 2*) *1*] **by** (*simp add*: *G1-def*)
have *fps-conv-radius G2* ≥ ∞
unfolding *G2-def* **by** (*intro conv-radius-fps-expansion holomorphic-intros*) *auto*
hence [*simp*]: *fps-conv-radius G2* = ∞
by *simp*
have *fps-conv-radius G2′* ≥ ∞
unfolding *G2′-def* **by** (*intro conv-radius-fps-expansion holomorphic-intros*) *auto*
hence [*simp*]: *fps-conv-radius G2′* = ∞
by *simp*
have [*simp*]: *fps-conv-radius G* = ∞

using *fps-conv-radius-mult*[*of G1 G2*] **by** (*simp add*: *G-def*)

have *eval-G1*: *eval-fps G1* (*s* − *1*) =
(*if s* = *1 then* −*ln 2 else* (*2 powr* (*1* − *s*) − *1*) / (*s* − *1*)) **for** *s*
unfolding *G1-def* **using** *fps-conv-radius-diff*[*of fps-exp* (−*Ln 2*) *1*]
by (*subst eval-fps-shift*)
(*auto intro*!: *subdegree-geI simp*: *eval-fps-diff powr-def exp-diff exp-minus algebra-simps*)
have *eval-G2*: *eval-fps G2* (*s* − *1*) = (*s* − *1*) ∗ *pre-zeta 1 s* + *1* **for** *s*
unfolding *G2-def* **by** (*subst eval-fps-expansion*[**where** *r* = ∞]) (*auto intro*!: *holomorphic-intros*)
have *eval-G*: *eval-fps G* (*s* − *1*) = *g s* **for** *s*
unfolding *G-def* **by** (*simp add*: *eval-fps-mult eval-G1 eval-G2 g-def perzeta-one-half-left′*)
have *eval-G′*: *eval-fps G s* = *g* (*1* + *s*) **for** *s*
using *eval-G*[*of s* + *1*] **by** (*simp add*: *add-ac*)
have *eval-G2′*: *eval-fps G2′* (*s* − *1*) = *pre-zeta 1 s* **for** *s*
unfolding *G2′-def* **by** (*intro eval-fps-expansion*[**where** *r* = ∞]) (*auto intro*!: *holomorphic-intros*)

show *G2* = *fps-X* ∗ *G2′* + *1*
proof (*intro eval-fps-eqD always-eventually allI*)
have ∗: *fps-conv-radius* (*fps-X* ∗ *G2′*) = ∞
using *fps-conv-radius-mult*[*of fps-X G2′*] **by** *simp*
from ∗ **show** *fps-conv-radius* (*fps-X* ∗ *G2′* + *1*) > *0*
using *fps-conv-radius-add*[*of fps-X* ∗ *G2′ 1*] **by** *auto*
show *eval-fps G2 s* = *eval-fps* (*fps-X* ∗ *G2′* + *1*) *s* **for** *s*
using ∗ *eval-G2*[*of 1* + *s*] *eval-G2′*[*of 1* + *s*]
by (*simp add*: *eval-fps-add eval-fps-mult*)
qed *auto*

have *G* = *fps-expansion g 1*
proof (*rule eval-fps-eqD*)
have *fps-conv-radius* (*fps-expansion g 1*) ≥ ∞
using *fraction-not-in-ints*[*of 2 1*]
by (*intro conv-radius-fps-expansion*) (*auto intro*!: *holomorphic-intros simp*: *g-def*)
thus *fps-conv-radius* (*fps-expansion g 1*) > *0* **by** *simp*
next
have *eval-fps* (*fps-expansion g 1*) *z* = *g* (*1* + *z*) **for** *z*
using *fraction-not-in-ints*[*of 2 1*]
by (*subst eval-fps-expansion′*[**where** *r* = ∞]) (*auto simp*: *g-def intro*!: *holomorphic-intros*)
thus *eventually* (λ*z*. *eval-fps G z* = *eval-fps* (*fps-expansion g 1*) *z*) (*nhds 0*)
by (*simp add*: *eval-G′*)
qed *auto*
thus (*deriv* ^^ *k*) *g 1* = *fact k* ∗ *fps-nth G k*
by (*simp add*: *fps-eq-iff fps-expansion-def*)
qed

lemma *stieltjes-gamma-aux4*: *fps-nth G k* = ($\sum$ *i*=*1*..*k*+*1*. (−*ln 2*)^*i* ∗ *A* (*k*−(*i*−*1*)) / *fact i*)
proof −
  have *fps-nth G k* = ($\sum$ *i*≤*k*. *fps-nth G1 i* ∗ *A* (*k* − *i*))
    unfolding *G-def fps-mult-nth A-def* **by** (*intro sum.cong*) *auto*
  also have ... = ($\sum$ *i*≤*k*. (−*ln 2*)^(*i*+*1*) ∗ *A* (*k* − *i*) / *fact* (*i*+*1*))
    by (*simp add*: *G1-def algebra-simps*)
  also have ... = ($\sum$ *i*=*1*..*k*+*1*. (−*ln 2*)^*i* ∗ *A* (*k*−(*i*−*1*)) / *fact i*)
    by (*intro sum.reindex-bij-witness*[*of* - λ*i*. *i*−*1 Suc*]) (*auto simp*: *Suc-diff-Suc*)
  finally show *?thesis* **.**
qed

lemma *stieltjes-gamma-aux5*: ($\sum$ *t*<*k*. (*k choose t*) ∗ *Ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*) −
      ln 2 ^ (*k*+*1*) / *of-nat* (*k*+*1*) = (−*1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*
proof −
  define *h* **where** *h* = (λ*k x*. ($\sum$ *n*=*1*..*x*. *ln*(*real n*)^*k* / *real n*) −
      ln (*real x*)^(*k*+*1*) / *real*(*k*+*1*) − *stieltjes-gamma k*)
  have *h-eq*: ($\sum$ *n*=*1*..*x*. *ln n* ^ *k* / *n*) = *ln x*^(*k*+*1*) / *real* (*k*+*1*) + *stieltjes-gamma k* + *h k x*
    for *k x* :: *nat* **by** (*simp add*: *h-def*)
  define *h′* **where** *h′* = (λ*x*. $\sum$ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *h t x*)
  define *S1* **where** *S1* = (λ*x*. ($\sum$ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *ln x* ^ (*t* + *1*) / (*t* + *1*)))
  define *S2* **where** *S2* = (λ*x*. ($\sum$ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *ln x* ^ (*t* + *1*) / (*k* + *1*)))

  have [*THEN filterlim-compose*, *tendsto-intros*]: *h t* ⟶ *0* **for** *t*
   using *tendsto-diff*[*OF stieltjes-gamma-real-limit-form*[*of t*] *tendsto-const*[*of stieltjes-gamma t*]]
    by (*simp add*: *h-def*)

  have *eq*: ($\sum$ *n*=*1*..*2* ∗ *x*. (−*1*)^(*n*+*1*) ∗ *ln n* ^ *k* / *n*) =
      ln 2 ^ (*k*+*1*) / *real* (*k*+*1*) −
      ($\sum$ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *stieltjes-gamma t*) + *h k* (*2*∗*x*) − *h′ x*
    (**is** *?lhs x* = *?rhs x*) **if** *x* > *0* **for** *x* :: *nat*
  proof −
    have *2* ∗ ($\sum$ *n*=*1*..*x*. *ln* (*2*∗*n*)^*k*/(*2*∗*n*)) =
        ($\sum$ *n*=*1*..*x*. $\sum$ *t*=*0*..*k*. *1*/*n* ∗ (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *ln n* ^ *t*)
      unfolding *sum-distrib-left*
    proof (*rule sum.cong*)
      fix *n* :: *nat* **assume** *n*: *n* ∈ {*1*..*x*}
      have *2* ∗ (*ln* (*2*∗*n*)^*k* / (*2*∗*n*)) = *1*/*n* ∗ (*ln n* + *ln 2*) ^ *k*
        using *n* **by** (*simp add*: *ln-mult add-ac*)
      also have (*ln n* + *ln 2*) ^ *k* = ($\sum$ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *ln n* ^ *t*)
        by (*subst binomial-ring*, *rule sum.cong*) *auto*
      also have *1*/*n* ∗ ... = ($\sum$ *t*=*0*..*k*. *1*/*n* ∗ (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *ln n*

^ t)
by (subst sum-distrib-left) (simp add: mult-ac)
finally show 2 ∗ (ln (2∗n)^k / (2∗n)) = ...
qed auto
also have . . . = (∑t=0..k. ∑n=1..x. 1/n ∗ (k choose t) ∗ ln 2 ^(k−t) ∗ ln n ^ t)
by (rule sum.swap)
also have . . . = (∑t=0..k. (k choose t) ∗ ln 2 ^(k − t) ∗
(ln x ^(t+1) / (t+1) + stieltjes-gamma t + h t x))
proof (rule sum.cong)
fix t :: nat assume t: t ∈ {0..k}
have (∑n=1..x. 1/n ∗ (k choose t) ∗ ln 2 ^(k−t) ∗ ln n ^ t) =
(k choose t) ∗ ln 2 ^(k − t) ∗ (∑n=1..x. ln n ^ t / n)
by (subst sum-distrib-left) (simp add: mult-ac)
also have (∑n=1..x. ln n ^ t / n) = ln x ^(t+1) / (t+1) + stieltjes-gamma t + h t x
using h-eq[of t] by simp
finally show (∑n=1..x. 1/n ∗ (k choose t) ∗ ln 2 ^(k−t) ∗ ln n ^ t) =
(k choose t) ∗ ln 2 ^(k − t) ∗ . . . .
qed simp-all
also have . . . = (∑t=0..k. (k choose t) / (t + 1) ∗ ln 2 ^(k − t) ∗ ln x ^(t + 1)) +
(∑t=0..k. (k choose t) ∗ ln 2 ^(k − t) ∗ stieltjes-gamma t) + h′ x
by (simp add: ring-distribs sum.distrib h′-def)
also have (∑t=0..k. (k choose t) / (t + 1) ∗ ln 2 ^(k − t) ∗ ln x ^(t + 1)) =
(∑t=0..k. (Suc k choose Suc t) / (k + 1) ∗ ln 2 ^(k − t) ∗ ln x ^(t + 1))
proof (intro sum.cong refl, goal-cases)
case (1 t)
have of-nat (k choose t) ∗ (of-nat (k + 1) :: real) = of-nat ((k choose t) ∗ (k + 1))
by (simp only: of-nat-mult)
also have (k choose t) ∗ (k + 1) = (Suc k choose Suc t) ∗ (t + 1)
using Suc-times-binomial-eq[of k t] by (simp add: algebra-simps)
also have of-nat . . . = of-nat (Suc k choose Suc t) ∗ (of-nat (t + 1) :: real)
by (simp only: of-nat-mult)
finally have ∗: of-nat (k choose t) / of-nat (t + 1) =
(of-nat (Suc k choose Suc t) / (k + 1) :: real)
by (simp add: divide-simps flip: of-nat-Suc del: binomial-Suc-Suc)
show ?case by (simp only: ∗)
qed
also have . . . = (∑t=1..Suc k. (Suc k choose t) / (k + 1) ∗ ln 2 ^(Suc k − t) ∗ ln x ^ t)
by (intro sum.reindex-bij-witness[of - λt. t−1 Suc]) auto
also have {1..Suc k} = {..Suc k} − {0} by auto
also have (∑t∈. . . . (Suc k choose t) / (k + 1) ∗ ln 2 ^(Suc k − t) ∗ ln x ^ t) =
(∑t≤Suc k. (Suc k choose t) / (k + 1) ∗ ln 2 ^(Suc k − t) ∗ ln x

^ *t*) −
ln 2 ^ *Suc k* / (*k* + *1*)
by (*subst sum-diff1*) *auto*
also have (∑ *t*≤*Suc k*. (*Suc k choose t*) / (*k* + *1*) ∗ *ln 2* ^ (*Suc k* − *t*) ∗ *ln x* ^ *t*) =
(*ln x* + *ln 2*) ^ *Suc k* / (*k* + *1*)
unfolding *binomial-ring* **by** (*subst sum-divide-distrib*) (*auto simp*: *algebra-simps*)
also have *ln x* + *ln 2* = *ln* (*2* ∗ *x*)
using ‹*x* > *0*› **by** (*simp add*: *ln-mult*)
finally have *eq1*: *2* ∗ (∑ *n*=*1*..*x*. *ln* (*real* (*2*∗*n*))^*k* / *real* (*2*∗*n*)) =
ln (*real* (*2*∗*x*))^(*k*+*1*) / *real* (*k*+*1*) − *ln 2*^(*k*+*1*) / *real* (*k*+*1*) +
(∑ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2*^(*k* − *t*) ∗ *stieltjes-gamma t*) + *h′ x*
by (*simp add*: *algebra-simps*)

have *eq2*: (∑ *n*=*1*..*2*∗*x*. *ln n* ^ *k* / *n*) = *ln* (*real* (*2*∗*x*))^(*k*+*1*) / *real* (*k*+*1*) + *stieltjes-gamma k* + *h k* (*2*∗*x*)
by (*simp only*: *h-eq*)

have (∑ *n*=*1*..*2*∗*x*. (−*1*)^(*n*+*1*) ∗ *ln n* ^ *k* / *n*) =
(∑ *n*=*1*..*2*∗*x*. *ln n* ^ *k* / *n* − *2* ∗ (*if even n then ln n* ^ *k* / *n else 0*))
by (*intro sum.cong*) *auto*
also have . . . = (∑ *n*=*1*..*2*∗*x*. *ln n* ^ *k* / *n*) −
2 ∗ (∑ *n*=*1*..*2*∗*x*. *if even n then ln n* ^ *k* / *n else 0*)
by (*simp only*: *sum-subtractf sum-distrib-left*)
also have (∑ *n*=*1*..*2*∗*x*. *if even n then ln n* ^ *k* / *n else 0*) =
(∑ *n* | *n* ∈ {*1*..*2*∗*x*} ∧ *even n*. *ln n* ^ *k* / *n*)
by (*intro sum.mono-neutral-cong-right*) *auto*
also have . . . = (∑ *n*=*1*..*x*. *ln* (*real* (*2*∗*n*)) ^ *k* / *real* (*2*∗*n*))
by (*intro sum.reindex-bij-witness*[*of* - λ*n*. *2*∗*n* λ*n*. *n div 2*]) *auto*
also have (∑ *n*=*1*..*2*∗*x*. *ln n* ^ *k* / *n*) − *2* ∗ . . . =
ln 2^(*k*+*1*) / *real* (*k*+*1*) −
((∑ *t*=*0*..*k*. (*k choose t*) ∗ *ln 2*^(*k* − *t*) ∗ *stieltjes-gamma t*) − *stieltjes-gamma k*) +
h k (*2*∗*x*) − *h′ x*
using *arg-cong2*[*OF eq1 eq2*, *of* (−)] **by** *simp*
also have {*0*..*k*} = *insert k* {..<*k*} **by** *auto*
also have (∑ *t*∈. . . . (*k choose t*) ∗ *ln 2*^(*k* − *t*) ∗ *stieltjes-gamma t*) − *stieltjes-gamma k* =
(∑ *t*<*k*. (*k choose t*) ∗ *ln 2*^(*k* − *t*) ∗ *stieltjes-gamma t*)
by (*subst sum.insert*) *auto*
finally show *?thesis* **.**
qed

have *?rhs* ⟶ *ln 2* ^ (*k*+*1*) / *real* (*k*+*1*) −
(∑ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *stieltjes-gamma t*)
unfolding *h′-def* **by** (*rule tendsto-eq-intros refl mult-nat-left-at-top filter-*

lim-ident | *simp*)+
  moreover have *eventually* (λx. *?rhs x* = *?lhs x*) *sequentially*
    using *eventually-gt-at-top*[*of 0*] **by** *eventually-elim* (*simp only*: *eq*)
  ultimately have ∗: *?lhs* ⟶ *ln 2* ^ (*k*+*1*) / *real* (*k*+*1*) −
      ($\sum$ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k*−*t*) ∗ *stieltjes-gamma t*)
    by (*rule Lim-transform-eventually*)
  also have (λx. $\sum$ *n*=*1*..*2*∗*x*. (−*1*) ^ (*n*+*1*) ∗ *ln* (*real n*) ^ *k* / *real n*) =
      (λx. $\sum$ *n*<*2*∗*x*. −((−*1*) ^ (*n*+*1*) ∗ *ln* (*real* (*n*+*1*)) ^ *k* / *real* (*n*+*1*)))
    by (*intro ext sum.reindex-bij-witness*[*of* - *Suc* λn. *n* − *1*]) (*auto simp*: *power-diff*)
  also have ... = (λx. −($\sum$ *n*<*2*∗*x*. ((−*1*) ^ (*n*+*1*) ∗ *ln* (*real* (*n*+*1*)) ^ *k* / *real* (*n*+*1*))))
    by (*subst sum-negf*) *auto*
  finally have ∗: ... ⟶ (*ln 2* ^ (*k*+*1*) / *real* (*k*+*1*) −
      ($\sum$ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*)) **.**
  have *lim1*: (λx. ($\sum$ *n*<*2*∗*x*. *complex-of-real* ((−*1*) ^ (*n*+*1*) ∗ *ln* (*real* (*n*+*1*)) ^ *k* / *real* (*n*+*1*))))
      ⟶ −(*ln 2* ^ (*k*+*1*) / *of-nat* (*k*+*1*) −
      ($\sum$ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*))
    (**is** *?lhs'* ⟶ -)
    using *tendsto-of-real*[*OF tendsto-minus*[*OF* ∗], **where** *?'a* = *complex*]
    by (*simp add*: *Ln-Reals-eq*)

  moreover have *?lhs'* ⟶ ((− *1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*)
  proof −
    have ∗∗: *filterlim* (λn::*nat*. *2* ∗ *n*) *sequentially sequentially* **by** *real-asymp*
    have (λx. ($\sum$ *n*<*2*∗*x*. *complex-of-real* ((−*1*) ^ (*n*+*1*) ∗ *ln* (*real* (*n*+*1*)) ^ *k* / *real* (*n*+*1*))))
      ⟶ ((− *1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*)
      by (*rule filterlim-compose*[*OF* - ∗∗]) (*use stieltjes-gamma-aux1* **in** ‹*simp add*: *sums-def*›)
    thus *?thesis* **.**
  qed

  ultimately have −(*ln 2* ^ (*k*+*1*) / *of-nat* (*k*+*1*) −
      ($\sum$ *t*<*k*. (*k choose t*) ∗ *ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*)) =
      (−*1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*
    by (*rule LIMSEQ-unique*)
  thus *?thesis* **by** (*simp add*: *Ln-Reals-eq*)
qed

lemma *stieltjes-gamma-aux6*: ($\sum$ *t*<*k*. (*k choose t*) ∗ *Ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*) −
      Ln 2 ^ (*k* + *1*) / *of-nat* (*k* + *1*) =
      (−*1*) ^ *k* ∗ *fact k* ∗ ($\sum$ *i*=*1*..*k*+*1*. (−*Ln 2*) ^ *i* ∗ *A* (*k*−(*i*−*1*)) / *fact i*)
proof −
  have ($\sum$ *t*<*k*. (*k choose t*) ∗ *Ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*) −
      Ln 2 ^ (*k* + *1*) / *of-nat* (*k* + *1*) = (− *1*) ^ *k* ∗ (*deriv* ^^ *k*) *g 1*
    using *stieltjes-gamma-aux5*[*of k*] **.**
  also have (*deriv* ^^ *k*) *g 1* = *fact k* ∗ *fps-nth G k*

by (*rule stieltjes-gamma-aux2*)
also have *fps-nth G k* = (∑ *i*=*1*..*k* + *1*. (−*Ln 2*) ^ *i* ∗ *A* (*k* − (*i* − *1*)) / *fact i*)
by (*rule stieltjes-gamma-aux4*)
finally show *?thesis* **by** (*simp add*: *mult-ac*)
qed

theorem *higher-deriv-pre-zeta-1-1*: (*deriv* ^^ *k*) (*pre-zeta 1*) *1* = (−*1*) ^ *k* ∗ *stieltjes-gamma k*
proof −
have *eq*: *A k* = (*if k* = *0 then 1 else* (−*1*)^(*k*+*1*) ∗ *stieltjes-gamma* (*k* − *1*) / *fact* (*k* − *1*)) **for** *k*
proof (*induction k rule*: *less-induct*)
case (*less k*)
show *?case*
proof (*cases k* = *0*)
case *True*
with *stieltjes-gamma-aux6*[*of 0*] **show** *?thesis* **by** *simp*
next
case *False*
have *k* ∗ *Ln 2* ∗ *stieltjes-gamma* (*k* − *1*) +
(∑ *t*<*k*−*1*. (*k choose t*) ∗ *Ln 2* ^(*k* − *t*) ∗ *stieltjes-gamma t*) =
(∑ *t*∈*insert* (*k*−*1*) {..<*k*−*1*}. (*k choose t*) ∗ *Ln 2* ^ (*k* − *t*) ∗ *stieltjes-gamma t*)
using *False* **by** (*subst sum.insert*) *auto*
also have *insert* (*k*−*1*) {..<*k*−*1*} = {..<*k*} **using** *False* **by** *auto*
also have (∑ *t*<*k*. *of-nat* (*k choose t*) ∗ *Ln 2* ^(*k* − *t*) ∗ *stieltjes-gamma t*) =
Ln 2 ^(*k* + *1*) / *of-nat* (*k* + *1*) +
(− *1*) ^ *k* ∗ *fact k* ∗ (∑ *i* = *1*..*k* + *1*. (− *Ln 2*) ^ *i* ∗ *A* (*k* − (*i* − *1*)) / *fact i*)
using *stieltjes-gamma-aux6*[*of k*] **by** (*simp add*: *algebra-simps*)
also have {*1*..*k*+*1*} = {*1*,*k*+*1*} ∪ {*2*..*k*} **by** *auto*
also have (− *1*) ^ *k* ∗ *fact k* ∗ (∑ *i*∈. . . . (− *Ln 2*) ^ *i* ∗ *A* (*k* − (*i* − *1*)) / *fact i*) =
(∑ *i*=*2*..*k*. (−*1*)^*k* ∗ *fact k* ∗ (− *Ln 2*) ^ *i* ∗ *A* (*k* − (*i* − *1*)) / *fact i*)
−*Ln 2* ∗ *A k* ∗ (− *1*) ^ *k* ∗ *fact k* +
(−*Ln 2*)^(*k*+*1*) ∗ *A 0* / *fact* (*k*+*1*) ∗ (− *1*) ^ *k* ∗ *fact k*
using *False* **by** (*subst sum.union-disjoint*)
(*auto simp*: *algebra-simps sum-distrib-left sum-distrib-right*)
also have (∑ *i*=*2*..*k*. (−*1*)^*k* ∗ *fact k* ∗ (−*Ln 2*) ^ *i* ∗ *A* (*k*−(*i*−*1*)) / *fact i*) =
(∑ *i*<*k*−*1*. (*k choose i*) ∗ *Ln 2* ^(*k*−*i*) ∗ *stieltjes-gamma i*)
using *False*
by (*intro sum.reindex-bij-witness*[*of* - λ*i*. *k* − *i* λ*i*. *k* − *i*])
(*auto simp*: *binomial-fact Suc-diff-le less field-simps power-neg-one-If*)
finally have *k* ∗ *Ln 2* ∗ *stieltjes-gamma* (*k* − *1*) =
(−*1*)^(*k*+*1*) ∗ *fact k* ∗ *Ln 2* ∗ *A k*

using *False* **by** (*simp add*: *less power-minus′*)
also have ... ∗ (−*1*)^(*k*+*1*) / *fact k* / *Ln 2* = *A k*
by *simp*
also have *k* ∗ *Ln 2* ∗ *stieltjes-gamma* (*k* − *1*) ∗ (−*1*)^(*k*+*1*) / *fact k* / *Ln 2* =
(−*1*)^(*k*+*1*) ∗ *stieltjes-gamma* (*k* − *1*) / *fact* (*k* − *1*)
using *False* **by** (*simp add*: *field-simps fact-reduce*)
finally have *A k* = (− *1*) ^ (*k* + *1*) ∗ *stieltjes-gamma* (*k* − *1*) / *fact* (*k* − *1*) **..**
thus *?thesis* **using** *False* **by** *simp*
qed
qed

have *fps-nth G2′ k* = *fps-nth G2* (*Suc k*)
by (*simp add*: *stieltjes-gamma-aux3*)
also have ... = *A* (*Suc k*)
by (*simp add*: *A-def*)
also have ... = (−*1*) ^ *k* ∗ *stieltjes-gamma k* / *fact k*
by (*simp add*: *eq*)
finally show (*deriv* ^^ *k*) (*pre-zeta 1*) *1* = (−*1*) ^ *k* ∗ *stieltjes-gamma k*
by (*simp add*: *G2′-def fps-eq-iff fps-expansion-def*)
qed

corollary *pre-zeta-1-1* [*simp*]: *pre-zeta 1 1* = *euler-mascheroni*
using *higher-deriv-pre-zeta-1-1*[*of 0*] **by** *simp*

corollary *zeta-minus-pole-limit*: (λ*s*. *zeta s* − *1* / (*s* − *1*)) −*1*→ *euler-mascheroni*
proof (*rule Lim-transform-eventually*)
show *eventually* (λ*s*. *pre-zeta 1 s* = *zeta s* − *1* / (*s* − *1*)) (*at 1*)
by (*auto simp*: *zeta-minus-pole-eq* [*symmetric*] *eventually-at-filter*)
have *isCont* (*pre-zeta 1*) *1*
by (*intro continuous-intros*) *auto*
thus *pre-zeta 1* −*1*→ *euler-mascheroni*
by (*simp add*: *isCont-def*)
qed

corollary *fps-expansion-pre-zeta-1-1*:
fps-expansion (*pre-zeta 1*) *1* = *Abs-fps* (λ*n*. (−*1*)^*n* ∗ *stieltjes-gamma n* / *fact n*)
by (*simp add*: *fps-expansion-def higher-deriv-pre-zeta-1-1*)

end

end

4 The Hadjicostas–Chapman formula

theory *Hadjicostas-Chapman*
imports *Zeta-Laurent-Expansion*

begin

In this section, we will derive a formula for the ζ function that was conjectured by Hadjicostas [4] and proven shortly afterwards by Chapman [3]. The formula is:

$$\begin{aligned}&\int_0^1 \int_0^1 \frac{(-\ln(xy))^z(1-x)}{1-xy}\,\mathrm{d}x\,\mathrm{d}y\\ &\quad= \int_0^1 \frac{(-\ln u)^z(-\ln u + u - 1)}{1-u}\,\mathrm{d}u\\ &\quad= \Gamma(z+2)\left(\zeta(z+2) - \frac{1}{z+1}\right)\end{aligned}$$

for any z with $\Re(z) > -2$. In particular, setting $z = 1$, we can derive the following formula for the Euler–Mascheroni constant γ:

$$-\int_0^1 \int_0^1 \frac{1-x}{(1-xy)\ln(xy)}\,\mathrm{d}x\,\mathrm{d}y = \gamma$$

This formula was first proven by Sondow [7].

4.1 The real case

We first define the integral for real $z > -2$. This is then a non-negative integral, so that we can ignore the issue of integrability and use the Lebesgue integral on the extended non-negative reals

We first show the equivalence of the single-integral and the double-integral form.

definition *Hadjicostas-nn-integral* :: *real* ⇒ *ennreal* **where**
  Hadjicostas-nn-integral z =
    set-nn-integral lborel {*0*<..<*1*}
      (λ*u*. *ennreal* ((−*ln u*) *powr z* / (*1* − *u*) ∗ (−*ln u* + *u* − *1*)))

definition *Hadjicostas-integral* :: *complex* ⇒ *complex* **where**
  Hadjicostas-integral z =
    (*LBINT u*=*0*..*1*. *of-real* (−*ln u*) *powr z* / *of-real* (*1* − *u*) ∗ *of-real* (−*ln u* + *u* − *1*))

lemma *Hadjicostas-nn-integral-altdef*:
  Hadjicostas-nn-integral z =
    (∫⁺(*x*,*y*)∈{*0*<..<*1*}×{*0*<..<*1*}. ((−*ln* (*x*∗*y*)) *powr z* ∗ (*1*−*x*) / (*1*−*x*∗*y*)) ∂*lborel*)
proof −
  define *f* **where** *f* ≡ (λ*u*. ((−*ln u*) *powr z* / (*1* − *u*) ∗ (−*ln u* + *u* − *1*)))
  let *?I* = *Gamma* (*z* + *2*) ∗ (*Re* (*zeta* (*z* + *2*)) − *1* / (*z* + *1*))
  let *?f* = λ*u*. ((−*ln u*) *powr z* / (*1* − *u*) ∗ (−*ln u* + *u* − *1*))
  define *D* :: (*real* × *real*) *set* **where** *D* = {*0*<..<*1*} × {*0*<..<*1*}

define $D1$ **where** $D1 = (SIGMA\ x{:}\{0{<}..{<}1\}.\ \{0{<}..{<}(x{::}real)\})$
define $D2$ **where** $D2 = (SIGMA\ u{:}\{0{<}..{<}1\}.\ \{u{<}..{<}(1{::}real)\})$
have [*measurable*]: $D1 \in sets\ (lborel \bigotimes_M lborel)$
proof −
have $D1 = \{x{\in}space\ (lborel \bigotimes_M lborel).\ snd\ x > 0 \wedge fst\ x > snd\ x \wedge fst\ x < 1\}$
by (*auto simp*: *D1-def space-pair-measure*)
also have $\ldots \in sets\ (lborel \bigotimes_M lborel)$
by *measurable*
finally show *?thesis* **.**
qed
have [*measurable*]: $D2 \in sets\ (lborel \bigotimes_M lborel)$
proof −
have $D2 = \{x{\in}space\ (lborel \bigotimes_M lborel).\ fst\ x > 0 \wedge fst\ x < snd\ x \wedge snd\ x < 1\}$
by (*auto simp*: *D2-def space-pair-measure*)
also have $\ldots \in sets\ (lborel \bigotimes_M lborel)$
by *measurable*
finally show *?thesis* **.**
qed

have $(\int^+(x,y){\in}D.\ ((-ln\ (x{*}y))\ powr\ z * (1{-}x)\ /\ (1{-}x{*}y))\ \partial lborel) =$
$(\int^+x{\in}\{0{<}..{<}1\}.\ (\int^+y{\in}\{0{<}..{<}1\}.\ ((-ln\ (x{*}y))\ powr\ z\ /\ (1{-}x{*}y) * (1{-}x))\ \partial lborel)\ \partial lborel)$
unfolding *lborel-prod* [*symmetric*] *case-prod-unfold D-def*
by (*subst lborel.nn-integral-fst*[*symmetric*])
(*auto intro*!: *nn-integral-cong simp*: *indicator-def*)
also have $\ldots = (\int^+x{\in}\{0{<}..{<}1\}.\ (\int^+u{\in}\{0{<}..{<}x\}.\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)\ \partial lborel)$
proof (*rule set-nn-integral-cong*)
fix x :: *real* **assume** x: $x \in space\ lborel \cap \{0{<}..{<}1\}$
show $(\int^+y{\in}\{0{<}..{<}1\}.\ ((-ln\ (x{*}y))\ powr\ z\ /\ (1{-}x{*}y) * (1{-}x))\ \partial lborel) =$
$(\int^+u{\in}\{0{<}..{<}x\}.\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)$ **using** x
by (*subst lborel-distr-mult'*[*of* $1/x$])
(*auto simp*: *nn-integral-density nn-integral-distr indicator-def field-simps*
simp flip: *ennreal-mult'* *intro*!: *nn-integral-cong*)
qed *auto*
also have $\ldots = (\int^+(x,u){\in}D1.\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)$
unfolding *lborel-prod* [*symmetric*] *case-prod-unfold D-def*
by (*subst lborel.nn-integral-fst*[*symmetric*], *measurable*)
(*auto intro*!: *nn-integral-cong simp*: *indicator-def D1-def*)
also have $\ldots = (\int^+(x,u).\ indicator\ D2\ (u,x) * ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)$
by (*intro nn-integral-cong*) (*auto simp*: *D1-def D2-def indicator-def split*: *if-splits*)
also have $\ldots = (\int^+u{\in}\{0{<}..{<}1\}.\ (\int^+x{\in}\{u{<}..{<}1\}.\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)\ \partial lborel)$

unfolding *case-prod-unfold lborel-prod* [*symmetric*]
by (*subst lborel-pair.nn-integral-snd* [*symmetric*], *measurable*)
(*auto intro*!: *nn-integral-cong simp*: *D2-def indicator-def*)
also have ... $= (\int^{+}u\in\{0<..<1\}.\ ((-ln\ u)\ powr\ z\ /\ (1 - u) * (-ln\ u + u - 1))\ \partial lborel)$
proof (*intro set-nn-integral-cong refl*)
fix u :: *real* **assume** u: $u \in space\ lborel \cap \{0<..<1\}$
let $?F = \lambda x.\ (-\ ln\ u)\ powr\ z\ /\ (1 - u) * (ln\ x - x)$
have $(\int^{+}x\in\{u<..<1\}.\ ennreal\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel) =$
$(\int^{+}x\in\{u..1\}.\ ennreal\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel)$
by (*rule nn-integral-cong-AE*, *rule AE-I*[*of - -* $\{u,1\}$])
(*auto simp*: *emeasure-lborel-countable indicator-def*)
also have ... $= ennreal\ (?F\ 1 - ?F\ u)$
using u **by** (*intro nn-integral-FTC-Icc*) (*auto intro*!: *derivative-eq-intros simp*: *divide-simps*)
also have $?F\ 1 - ?F\ u = (-ln\ u)\ powr\ z\ /\ (1 - u) * (-ln\ u + u - 1)$
using u **by** (*simp add*: *divide-simps*) (*simp add*: *algebra-simps*)?
finally show $(\int^{+}x\in\{u<..<1\}.\ ((-\ ln\ u)\ powr\ z\ /\ (1 - u) * (1 - x)\ /\ x)\ \partial lborel) = ennreal$ ... **.**
qed
also have ... = *Hadjicostas-nn-integral z*
by (*simp add*: *Hadjicostas-nn-integral-def*)
finally show *?thesis* **by** (*simp add*: *D-def*)
qed

We now solve the single integral for real $z > -1$.

lemma *Hadjicostas-Chapman-aux*:
fixes z :: *real*
assumes z: $z > -1$
defines $f \equiv (\lambda u.\ ((-ln\ u)\ powr\ z\ /\ (1 - u) * (-ln\ u + u - 1)))$
shows (f *has-integral* $(Gamma\ (z + 2) * (Re\ (zeta\ (z + 2)) - 1\ /\ (z + 1))))$ $\{0<..<1\}$
proof −
let $?I = Gamma\ (z + 2) * (Re\ (zeta\ (z + 2)) - 1\ /\ (z + 1))$
have *nonneg*: $1 \le x + exp\ (-\ x)$ **if** $x \ge 0$ **for** x :: *real*
proof −
have $x + (1 + (-x)) \le x + exp\ (-x)$
by (*intro add-left-mono exp-ge-add-one-self*)
thus *?thesis* **by** *simp*
qed

have *eq*: $((\lambda t{::}real.\ exp\ (-t))\ \text{`}\ \{0<..\}) = \{0<..<1\}$
proof *safe*
fix x :: *real* **assume** x: $x \in \{0<..<1\}$
hence $x = exp\ (-(-ln\ x))$ **and** $-ln\ x \in \{0<..\}$
by *auto*
thus $x \in (\lambda t.\ exp\ (-t))\ \text{`}\ \{0<..\}$ **by** *blast*
qed *auto*

have *I*: $((\lambda x.\ x\ powr\ (z{+}1)\ /\ (exp\ x\ -\ 1)\ -\ x\ powr\ z\ /\ exp\ x)$ *has-integral* *?I*) $\{0{<}..\}$
proof −
from *z* **have** $z + 1 \notin \mathbb{R}_{\leq 0}$
by (*auto simp*: *nonpos-Reals-def*)
hence *z′*: $z + 1 \notin \mathbb{Z}_{\leq 0}$
using *nonpos-Ints-subset-nonpos-Reals* **by** *blast*
have $((\lambda x.\ x\ powr\ (z + 2 - 1)\ /\ (exp\ x - 1) - x\ powr\ (z + 1 - 1)\ /\ exp\ x)$
has-integral $(Gamma\ (z + 2) * Re\ (zeta\ (z + 2)) - Gamma\ (z + 1)))$
$\{0{<}..\}$ **using** *z*
by (*intro has-integral-diff Gamma-integral-real′ Gamma-times-zeta-has-integral-real*)
auto
also have $Gamma\ (z + 2) * Re\ (zeta\ (z + 2)) - Gamma\ (z + 1) =$
$Gamma\ (z + 2) * (Re\ (zeta\ (z + 2)) - 1\ /\ (z + 1))$
using *Gamma-plus1*[*of z+1*] *z z′* **by** (*auto simp*: *field-simps*)
finally show *?thesis*
by (*simp add*: *add-ac*)
qed
also have *?this* $\longleftrightarrow ((\lambda x.\ |-exp\ (-x)| * f\ (exp\ (-x)))$ *has-integral* *?I*) $\{0{<}..\}$
unfolding *f-def*
apply (*intro has-integral-cong*)
apply (*auto simp*: *field-simps powr-add powr-def exp-add*)
apply (*simp flip*: *exp-add*)
done
finally have ∗: $((\lambda x.\ |-exp\ (-x)| * f\ (exp\ (-x)))$ *has-integral* *?I*) $\{0{<}..\}$ **.**

have $((\lambda x.\ |-exp\ (-x)| *_R f\ (exp\ (-x)))$ *absolutely-integrable-on* $\{0{<}..\}) \wedge$
integral $\{0{<}..\}\ (\lambda x.\ |-exp\ (-x)| *_R f\ (exp\ (-x))) = ?I$
proof (*intro conjI nonnegative-absolutely-integrable-1*)
fix *x* :: *real* **assume** *x*: $x \in \{0{<}..\}$
thus $|-exp\ (-x)| *_R f\ (exp\ (-x)) \geq 0$
unfolding *f-def* **using** *nonneg*
by (*intro scaleR-nonneg-nonneg mult-nonneg-nonneg divide-nonneg-nonneg*)
auto
qed (*use* ∗ **in** ‹*simp-all add*: *has-integral-iff*›)

also have *?this* $\longleftrightarrow$ *f absolutely-integrable-on* $(\lambda x.\ exp\ (-\ x))$ ‘ $\{0{<}..\} \wedge$
integral $((\lambda x.\ exp\ (-\ x))$ ‘ $\{0{<}..\})\ f = ?I$
by (*intro has-absolute-integral-change-of-variables-1′*)
(*auto intro*!: *derivative-eq-intros inj-onI*)
also have $(\lambda x{::}real.\ exp\ (-\ x))$ ‘ $\{0{<}..\} = \{0{<}..{<}1\}$
by (*fact eq*)
finally show (*f has-integral ?I*) $\{0{<}..{<}1\}$
by (*auto simp*: *has-integral-iff dest*: *set-lebesgue-integral-eq-integral*)
qed

lemma *real-zeta-ge-one-over-minus-one*:
fixes *z* :: *real*

assumes *z*: $z > 1$
shows *Re* (*zeta* (*complex-of-real z*)) $\geq 1 / (z - 1)$
proof −
have *ineq*: $1 \leq x - ln\ x$ **if** $x \in \{0<..<1\}$ **for** *x* :: *real*
using *ln-le-minus-one*[*of x*] *that* **by** *simp*
have ∗: $((\lambda u.\ (-\ ln\ u)\ powr\ (z - 2) * (u - ln\ u - 1) / (1 - u))$ *has-integral*
Gamma z ∗ (*Re* (*zeta* (*complex-of-real z*)) $- 1 / (z - 1)$)) $\{0<..<1\}$
using *Hadjicostas-Chapman-aux*[*of z* − *2*] *z* **by** *simp*
from *ineq* **have** *Gamma z* ∗ (*Re* (*zeta* (*complex-of-real z*)) $- 1 / (z - 1)) \geq 0$
by (*intro has-integral-nonneg*[*OF* ∗] *z mult-nonneg-nonneg divide-nonneg-nonneg*)
auto
moreover have *Gamma z* > *0*
using *assms* **by** (*intro Gamma-real-pos*) *auto*
ultimately show *Re* (*zeta* (*complex-of-real z*)) $\geq 1 / (z - 1)$
by (*subst* (*asm*) *zero-le-mult-iff*) *auto*
qed

We now have the formula for real $z > -1$.

lemma *Hadjicostas-Chapman-formula-real*:
fixes *z* :: *real*
assumes *z*: $z > -1$
shows *Hadjicostas-nn-integral z* =
ennreal (*Gamma* (*z* + *2*) ∗ (*Re* (*zeta* (*z* + *2*)) $- 1 / (z + 1)$)))
proof −
have *nonneg*: $1 \leq x - ln\ x$ **if** $x > 0$ $x < 1$ **for** *x* :: *real*
proof −
have $ln\ x + (1 + ln\ x) \leq ln\ x + exp\ (ln\ x)$
by (*intro add-left-mono exp-ge-add-one-self*)
thus *?thesis* **using** *that* **by** (*simp add*: *exp-minus*)
qed
show *?thesis*
unfolding *Hadjicostas-nn-integral-def* **using** *nonneg Hadjicostas-Chapman-aux*[*OF*
z]
by (*intro nn-integral-has-integral-lebesgue′ mult-nonneg-nonneg divide-nonneg-nonneg*)
auto
qed

4.2 Analyticity of the integral

To extend the formula to its full domain of validity (any complex z with $\Re(z) > -2$), we will use analytic continuation. To do this, we first have to show that the integral is an analytic function of z on that domain. This is unfortunately somewhat involved, since the integral is an improper one and we first need to show uniform convergence so that we can pull the derivative inside the integral sign.

We will use the single-integral form so that we only have to deal with one integral and not two.

context

```
  fixes f :: complex ⇒ real ⇒ complex
  defines f ≡ (λz u. of-real (−ln u) powr z / of-real (1 − u) ∗ of-real (−ln u +
u − 1))
begin

context
  fixes x y :: real and g1 g2 :: real ⇒ real
  assumes x > −2
  defines g1 ≡ (λx. (− ln x) powr y ∗ (x − ln x − 1) / (1 − x))
  defines g2 ≡ (λu. (−ln u) powr x ∗ (u − ln u − 1) / (1 − u))
begin

lemma integrable-bound1:
  interval-lebesgue-integrable lborel 0 (ereal (exp (− 1))) g1
  unfolding zero-ereal-def
proof (rule interval-lebesgue-integrable-bigo-left)
  show g1 ∈ O[at-right 0](λu. u powr (−1/2))
    unfolding g1-def by real-asymp
  show continuous-on {0<..exp(−1)} g1
    unfolding g1-def by (auto intro!: continuous-intros)
  have set-integrable lborel (einterval 0 (exp (−1))) (λu. u powr (−1/2))
  proof (rule interval-integral-FTC-nonneg)
    fix u :: real assume u: 0 < ereal u ereal u < ereal (exp (−1))
    show ((λu. 2 ∗ u powr (1/2)) has-field-derivative (u powr (−1/2))) (at u)
      using u by (auto intro!: derivative-eq-intros simp: power2-eq-square)
    show isCont (λu. u powr (−1/2)) u
      using u by (auto intro!: continuous-intros)
  next
   show (((λu. 2 ∗ u powr (1/2)) ∘ real-of-ereal) ⟶ 2 ∗ exp (−1) powr (1/2))
         (at-left (ereal (exp (− 1))))
      unfolding ereal-tendsto-simps by real-asymp
    show (((λu. 2 ∗ u powr (1/2)) ∘ real-of-ereal) ⟶ 0) (at-right 0)
      unfolding zero-ereal-def unfolding ereal-tendsto-simps by real-asymp
  qed auto
  thus interval-lebesgue-integrable lborel (ereal 0) (ereal (exp (− 1)))
        (λu. u powr (−1/2))
    by (simp add: interval-lebesgue-integrable-def zero-ereal-def)
qed (auto simp add: g1-def set-borel-measurable-def)

lemma integrable-bound2:
  interval-lebesgue-integrable lborel (exp (−1)) 1 g2
      unfolding one-ereal-def
proof (rule interval-lebesgue-integrable-bigo-right)
  show g2 ∈ O[at-left 1](λu. (1 − u) powr (x + 1))
    unfolding g2-def by real-asymp
  have ln x ≠ 0 if x ∈ {exp (−1)..<1} for x :: real
  proof −
    have 0 < exp (−1 :: real) by simp
    also have ... ≤ x using that by auto
```

finally have *x* > *0* **.**
from *that* ‹*x* > *0*› **have** *ln x* < *ln 1*
by (*subst ln-less-cancel-iff*) *auto*
thus *ln x* ≠ *0* **by** *simp*
qed
thus *continuous-on* {*exp* (− *1*)..<*1*} *g2*
unfolding *g2-def* **by** (*auto intro*!: *continuous-intros*)
let *?F* = (λ*u*. −*1* / (*x* + *2*) ∗ (*1* − *u*) *powr* (*x* + *2*))
have *set-integrable lborel* (*einterval* (*exp* (−*1*)) *1*) (λ*u*. (*1* − *u*) *powr* (*x* + *1*))
proof (*rule interval-integral-FTC-nonneg*[**where** *F* = *?F*])
fix *u* :: *real* **assume** *u*: *ereal* (*exp* (−*1*)) < *ereal u ereal u* < *1*
show (*?F has-field-derivative* (*1* − *u*) *powr* (*x* + *1*)) (*at u*)
using *u* ‹*x* > −*2*› **by** (*auto intro*!: *derivative-eq-intros simp*: *one-ereal-def add-ac*)
show *isCont* (λ*u*. (*1* − *u*) *powr* (*x* + *1*)) *u*
using *u* **by** (*auto intro*!: *continuous-intros*)
next
show (((λ*u*. − *1* / (*x* + *2*) ∗ (*1* − *u*) *powr* (*x* + *2*)) ∘ *real-of-ereal*) ⟶
− *1* / (*x* + *2*) ∗ (*1* − *exp* (−*1*)) *powr* (*x* + *2*)) (*at-right* (*ereal* (*exp* (− *1*))))
unfolding *ereal-tendsto-simps* **by** *real-asymp*
show (((λ*u*. − *1* / (*x* + *2*) ∗ (*1* − *u*) *powr* (*x* + *2*)) ∘ *real-of-ereal*) ⟶ *0*) (*at-left 1*)
unfolding *one-ereal-def* **unfolding** *ereal-tendsto-simps*
using ‹*x* > −*2*› **by** *real-asymp*
qed *auto*
thus *interval-lebesgue-integrable lborel* (*ereal* (*exp* (− *1*)))
(*ereal 1*) (λ*u*. (*1* − *u*) *powr* (*x* + *1*))
by (*simp add*: *interval-lebesgue-integrable-def one-ereal-def*)
qed (*auto simp add*: *g2-def set-borel-measurable-def*)

lemma *bound2*:
norm (*f z u*) ≤ *g2 u* **if** *z*: *Re z* ∈ {*x*..*y*} **and** *u*: *u* ∈ {*exp* (−*1*)<..<*1*} **for** *z u*
proof −
have *0* < *exp* (−*1*::*real*) **by** *simp*
also have ... ≤ *u* **using** *u* **by** (*simp add*: *einterval-def*)
finally have *u* > *0* **.**

from *u* ‹*u* > *0*› **have** *ln-u*: *ln u* > *ln* (*exp* (−*1*))
by (*subst ln-less-cancel-iff*) (*auto simp*: *einterval-def*)
from *z u* ‹*u* > *0*› **have** *norm* (*f z u*) = (− *ln u*) *powr Re z* ∗ |*u* − *ln u* − *1*| / (*1* − *u*)
unfolding *f-def norm-mult norm-divide norm-of-real*
by (*simp add*: *norm-powr-real-powr einterval-def*)
also have |*u* − *ln u* − *1*| = *u* − *ln u* − *1*
using *u* ‹*u* > *0*› *ln-add-one-self-le-self2*[*of u* − *1*] **by** (*simp add*: *einterval-def*)
also have (− *ln u*) *powr Re z* ∗ (*u* − *ln u* − *1*) / (*1* − *u*) ≤
(− *ln u*) *powr x* ∗ (*u* − *ln u* − *1*) / (*1* − *u*)
using *z u* ‹*u* > *0*› *ln-u ln-add-one-self-le-self2*[*of u* − *1*]

by (*intro mult-right-mono divide-right-mono powr-mono'*) (*auto simp*: *einterval-def*)
finally show *norm* (*f z u*) $\leq$ *g2 u* **by** (*simp add*: *g2-def*)
qed

lemma *integrable2-aux*: *interval-lebesgue-integrable lborel* (*exp* ($-$*1*)) *1* (*f z*)
if *z*: *Re z* $\in$ {*x..y*} **for** *z*
proof $-$
have *set-integrable lborel* {*exp* ($-$*1*)<..<*1*} (*f z*)
proof (*rule set-integrable-bound*[*OF - - AE-I2*[*OF impI*]])
fix *u* :: *real* **assume** *u* $\in$ {*exp* ($-$*1*)<..<*1*}
hence *norm* (*f z u*) $\leq$ *g2 u* **using** *z* **by** (*intro bound2*) *auto*
also have ... $\leq$ *norm* (*g2 u*) **by** *simp*
finally show *norm* (*f z u*) $\leq$ *norm* (*g2 u*) **.**
qed (*use integrable-bound2* **in** ‹*simp-all add*: *interval-lebesgue-integrable-def one-ereal-def set-borel-measurable-def f-def*›)
thus *?thesis* **by** (*simp add*: *interval-lebesgue-integrable-def one-ereal-def*)
qed

lemma *uniform-limit2*:
uniform-limit {*z. Re z* $\in$ {*x..y*}}
(λ*a z. LBINT u=exp* ($-$*1*)*..a. f z u*)
(λ*z. LBINT u=exp* ($-$*1*)*..1. f z u*) (*at-left 1*)
by (*intro uniform-limit-interval-integral-right*[*of - - g2*] *AE-I2 impI*)
(*use bound2 integrable-bound2* **in** ‹*simp-all add*: *einterval-def f-def set-borel-measurable-def*›)

lemma *uniform-limit2'*:
uniform-limit {*z. Re z* $\in$ {*x..y*}}
(λ*n z. LBINT u=exp* ($-$*1*)*..ereal* (*1*$-$(*1*/*2*)^*Suc n*)*. f z u*)
(λ*z. LBINT u=exp* ($-$*1*)*..1. f z u*) *sequentially*
proof (*rule filterlim-compose*[*OF uniform-limit2*])
have *filterlim* (λ*n. 1* $-$ (*1*/*2*)^*Suc n* :: *real*) (*at-left 1*) *sequentially*
by *real-asymp*
hence *filtermap ereal* (*filtermap* (λ*n.* (*1* $-$ (*1* / *2*) ^ *Suc n*)) *sequentially*) $\leq$
filtermap ereal (*at-left 1*)
unfolding *filterlim-def* **by** (*rule filtermap-mono*)
thus *filterlim* (λ*n. ereal* (*1* $-$ (*1*/*2*)^*Suc n*)) (*at-left 1*) *sequentially*
unfolding *one-ereal-def at-left-ereal* **by** (*simp add*: *filterlim-def filtermap-filtermap*)
qed

lemma *bound1*: *norm* (*f z u*) $\leq$ *g1 u* **if** *z*: *Re z* $\in$ {*x..y*} **and** *u*: *u* $\in$ {*0*<..<*exp* ($-$*1*)} **for** *z u*
proof $-$
from *u* **have** *u* $\leq$ *exp* ($-$*1*) **by** (*simp add*: *einterval-def*)
also have *exp* ($-$*1*) < *exp* (*0*::*real*)
by (*subst exp-less-cancel-iff*) *auto*
also have *exp* (*0*::*real*) = *1* **by** *simp*
finally have *u* < *1* **.**
from *u* **have** *ln u* < *ln* (*exp* ($-$*1*))

by (*subst ln-less-cancel-iff*) (*auto simp*: *einterval-def*)
hence *ln-u*: *ln u* $<$ *−1* **by** *simp*
from *z u* ‹*u* $<$ *1*› **have** *norm* (*f z u*) = (− *ln u*) *powr Re z* ∗ |*u* − *ln u* − *1*| / (*1* − *u*)
unfolding *f-def norm-mult norm-divide norm-of-real*
by (*simp add*: *norm-powr-real-powr einterval-def*)
also have |*u* − *ln u* − *1*| = *u* − *ln u* − *1*
using *u ln-add-one-self-le-self2*[*of u* − *1*] **by** (*simp add*: *einterval-def*)
also have (− *ln u*) *powr Re z* ∗ (*u* − *ln u* − *1*) / (*1* − *u*) ≤
(− *ln u*) *powr y* ∗ (*u* − *ln u* − *1*) / (*1* − *u*)
using *z u ln-u* ‹*u* $<$ *1*›
by (*intro mult-right-mono divide-right-mono powr-mono*) (*auto simp*: *einterval-def*)
finally show *norm* (*f z u*) ≤ *g1 u* **by** (*simp add*: *g1-def*)
qed

lemma *integrable1-aux*: *interval-lebesgue-integrable lborel 0* (*exp* (*−1*)) (*f z*)
if *z*: *Re z* ∈ {*x*..*y*} **for** *z*
proof −
have *set-integrable lborel* {*0*$<$..$<$*exp* (*−1*)} (*f z*)
proof (*rule set-integrable-bound*[*OF* - - *AE-I2*[*OF impI*]])
fix *u* :: *real* **assume** *u* ∈ {*0*$<$..$<$*exp* (*−1*)}
hence *norm* (*f z u*) ≤ *g1 u* **using** *z* **by** (*intro bound1*) *auto*
also have . . . ≤ *norm* (*g1 u*) **by** *simp*
finally show *norm* (*f z u*) ≤ *norm* (*g1 u*) **.**
qed (*use integrable-bound1* **in** ‹*simp-all add*: *interval-lebesgue-integrable-def zero-ereal-def set-borel-measurable-def f-def*›)
thus *?thesis* **by** (*simp add*: *interval-lebesgue-integrable-def zero-ereal-def*)
qed

lemma *uniform-limit1*:
uniform-limit {*z*. *Re z* ∈ {*x*..*y*}}
(λ*a z*. *LBINT u*=*a*..*exp* (*−1*). *f z u*)
(λ*z*. *LBINT u*=*0*..*exp* (*−1*). *f z u*) (*at-right 0*)
by (*intro uniform-limit-interval-integral-left*[*of* - - *g1*] *AE-I2 impI*)
(*use bound1 integrable-bound1* **in** ‹*simp-all add*: *einterval-def f-def set-borel-measurable-def*›)

lemma *uniform-limit1′*:
uniform-limit {*z*. *Re z* ∈ {*x*..*y*}}
(λ*n z*. *LBINT u*=*ereal* ((*1*/*2*)^*Suc n*)..*exp* (*−1*). *f z u*)
(λ*z*. *LBINT u*=*0*..*exp* (*−1*). *f z u*) *sequentially*
proof (*rule filterlim-compose*[*OF uniform-limit1*])
have *filterlim* (λ*n*. (*1*/*2*)^*Suc n* :: *real*) (*at-right 0*) *sequentially*
by *real-asymp*
hence *filtermap ereal* (*filtermap* (λ*n*. ((*1* / *2*) ^ *Suc n*)) *sequentially*) ≤
filtermap ereal (*at-right 0*)
unfolding *filterlim-def* **by** (*rule filtermap-mono*)
thus *filterlim* (λ*n*. *ereal* ((*1*/*2*)^*Suc n*)) (*at-right 0*) *sequentially*
unfolding *zero-ereal-def at-right-ereal* **by** (*simp add*: *filterlim-def filtermap-filtermap*)

qed

end

With all of the above bounds, we have shown that the integral exists for any z with $\Re(z) > -2$.

theorem *Hadjicostas-integral-integrable*: *interval-lebesgue-integrable lborel 0 1* (*f z*)
  if *z*: *Re z* > −*2*
proof −
  from *dense*[*OF z*] **obtain** *x* **where** *x*: *x* > −*2 Re z* > *x* **by** *blast*
  have *interval-lebesgue-integrable lborel 0* (*exp*(−*1*)) (*f z*)
    by (*rule integrable1-aux*[*of x - Re z* + *1*]) (*use x* **in** *auto*)
  moreover have *interval-lebesgue-integrable lborel* (*exp*(−*1*)) *1* (*f z*)
    by (*rule integrable2-aux*[*of x - Re z* + *1*]) (*use x* **in** *auto*)
  ultimately show *interval-lebesgue-integrable lborel 0 1* (*f z*)
   by (*rule interval-lebesgue-integrable-combine*) (*auto simp*: *f-def set-borel-measurable-def*)
qed

lemma *integral-holo-aux*:
  assumes *ab*: *a* > *0 a* ≤ *b b* < *1*
  shows (*λz. LBINT u=ereal a..ereal b. f z u*) *holomorphic-on A*
proof −
  define *f′* :: *complex* ⇒ *real* ⇒ *complex*
    where *f′* ≡ (*λz u. ln* (−*ln u*) ∗ *f z u*)
  note [*derivative-intros*] = *has-field-derivative-complex-powr-right′*

  have (*λz. integral* (*cbox a b*) (*f z*)) *holomorphic-on UNIV*
  proof (*rule leibniz-rule-holomorphic*[*of - - - - f′*], *goal-cases*)
    case (*1 z t*)
    show *?case* **unfolding** *f-def*
      apply (*insert 1 ab*)
      apply (*rule derivative-eq-intros refl* | *simp*)+
      apply (*auto simp*: *f′-def field-simps f-def Ln-of-real*)
      done
  next
    from *ab* **show** *continuous-on* (*UNIV* × *cbox a b*) (*λ*(*z*, *t*). *f′ z t*)
     by (*auto simp*: *case-prod-unfold f′-def f-def Ln-of-real intro*!: *continuous-intros*)
  next
    fix *z* :: *complex*
    show *f z integrable-on cbox a b*
      unfolding *f-def f′-def* **using** *ab*
      by (*intro integrable-continuous continuous-intros*) *auto*
  qed (*auto simp*: *convex-halfspace-Re-gt*)
  also have (*λz. integral* (*cbox a b*) (*f z*)) = (*λz.* ∫ *u*∈*cbox a b. f z u* ∂*lborel*)
  proof (*intro ext set-borel-integral-eq-integral*(*2*) [*symmetric*])
    fix *z* :: *complex*
    show *complex-set-integrable lborel* (*cbox a b*) (*f z*)
      unfolding *f-def* **using** *ab*

```
    by (intro continuous-on-imp-set-integrable-cbox continuous-intros) (auto simp:
Ln-of-real)
  qed
  also have ... = (λz. LBINT u=a..b. f z u)
    using ab by (simp add: interval-integral-Icc)
  finally show ?thesis by (rule holomorphic-on-subset) auto
qed

lemma integral-holo:
  assumes ab: min a b > 0 max a b < 1
  shows (λz. LBINT u=ereal a..ereal b. f z u) holomorphic-on A
proof (cases a ≤ b)
  case True
  thus ?thesis using assms integral-holo-aux[of a b] by auto
next
  case False
  have (λz. −(LBINT u=ereal b..ereal a. f z u)) holomorphic-on A
    using False assms by (intro holomorphic-intros integral-holo-aux) auto
  thus ?thesis by (subst interval-integral-endpoints-reverse)
qed

lemma holo1: (λz. LBINT u=0..exp (−1). f z u) holomorphic-on {z. Re z > −2}
proof (rule holomorphic-uniform-sequence
    [where f = (λn z. LBINT u=ereal ((1/2)^Suc n)..exp (−1). f z u)], goal-cases)
  case (3 z)
  define ε where ε = (Re z + 2) / 2
  from 3 have ε > 0 by (auto simp: ε-def)
  have subset: cball z ε ⊆ {s. Re s ∈ {Re z − ε..Re z + ε}}
  proof safe
    fix s assume s: s ∈ cball z ε
    have |Re (s − z)| ≤ norm (s − z) by (rule abs-Re-le-cmod)
    also have ... ≤ ε using s by (simp add: dist-norm norm-minus-commute)
    finally show Re s ∈ {Re z − ε..Re z + ε} by auto
  qed

  show ?case
  proof (rule exI[of - ε], intro conjI)
    have cball z ε ⊆ {s. Re s ∈ {Re z − ε..Re z + ε}} by fact
    also have ... ⊆ {s. Re s > −2}
      using 3 by (auto simp: ε-def field-simps)
    finally show cball z ε ⊆ {s. Re s > −2} .
  next
    from 3 have Re z − ε > −2 by (simp add: ε-def field-simps)
    thus uniform-limit (cball z ε) (λn z. LBINT u=ereal ((1 / 2) ^ Suc n)..ereal
(exp (− 1)). f z u)
          (λz. LBINT u=0..ereal (exp(−1)). f z u) sequentially
      using uniform-limit-on-subset[OF uniform-limit1' subset] by simp
  qed fact+
next
```

fix *n* :: *nat*
have $(1\ /\ 2)$ ^ *Suc n* $<$ $(1\ /\ 2$ :: *real*$)$ ^ 0
by (*subst power-strict-decreasing-iff*) *auto*
thus $(\lambda z.\ LBINT\ u{=}ereal\ ((1\ /\ 2)$ ^ $Suc\ n)..ereal\ (exp\ (-\ 1)).\ f\ z\ u)$ *holomorphic-on* $\{z.\ Re\ z > -2\}$
by (*intro integral-holo*) *auto*
qed (*auto simp*: *open-halfspace-Re-gt*)

lemma *holo2*: $(\lambda z.\ LBINT\ u{=}exp\ (-1)..1.\ f\ z\ u)$ *holomorphic-on* $\{z.\ Re\ z > -2\}$
proof (*rule holomorphic-uniform-sequence*
[**where** $f = (\lambda n\ z.\ LBINT\ u{=}exp\ (-1)..ereal\ (1-(1/2)$^$Suc\ n).\ f\ z\ u)$], *goal-cases*)
case (*3 z*)
define ε **where** $\varepsilon = (Re\ z + 2)\ /\ 2$
from *3* **have** $\varepsilon > 0$ **by** (*auto simp*: ε-*def*)
have *subset*: $cball\ z\ \varepsilon \subseteq \{s.\ Re\ s \in \{Re\ z - \varepsilon..Re\ z + \varepsilon\}\}$
proof *safe*
fix *s* **assume** *s*: $s \in cball\ z\ \varepsilon$
have $|Re\ (s - z)| \leq norm\ (s - z)$ **by** (*rule abs-Re-le-cmod*)
also have $\ldots \leq \varepsilon$ **using** *s* **by** (*simp add*: *dist-norm norm-minus-commute*)
finally show $Re\ s \in \{Re\ z - \varepsilon..Re\ z + \varepsilon\}$ **by** *auto*
qed

show *?case*
proof (*rule exI*[*of* - ε], *intro conjI*)
have $cball\ z\ \varepsilon \subseteq \{s.\ Re\ s \in \{Re\ z - \varepsilon..Re\ z + \varepsilon\}\}$ **by** *fact*
also have $\ldots \subseteq \{s.\ Re\ s > -2\}$
using *3* **by** (*auto simp*: ε-*def field-simps*)
finally show $cball\ z\ \varepsilon \subseteq \{s.\ Re\ s > -2\}$ **.**
next
from *3* **have** $Re\ z - \varepsilon > -2$ **by** (*simp add*: ε-*def field-simps*)
thus *uniform-limit* $(cball\ z\ \varepsilon)\ (\lambda n\ z.\ LBINT\ u{=}ereal\ (exp\ (-\ 1))..ereal\ (1-(1/2)$^$Suc\ n).\ f\ z\ u)$
$(\lambda z.\ LBINT\ u{=}ereal\ (exp(-1))..1.\ f\ z\ u)$ *sequentially*
using *uniform-limit-on-subset*[*OF uniform-limit2' subset*] **by** *simp*
qed *fact+*
next
fix *n* :: *nat*
have $(1\ /\ 2)$ ^ *Suc n* $<$ $(1\ /\ 2$ :: *real*$)$ ^ 0
by (*subst power-strict-decreasing-iff*) *auto*
thus $(\lambda z.\ LBINT\ u{=}ereal\ (exp\ (-1))..ereal\ (1-(1/2)$^$Suc\ n).\ f\ z\ u)$ *holomorphic-on* $\{z.\ Re\ z > -2\}$
by (*intro integral-holo*) *auto*
qed (*auto simp*: *open-halfspace-Re-gt*)

Finally, we have shown that Hadjicostas's integral is an analytic function of z in the domain $\Re(z) > -2$.

lemma *holomorphic-Hadjicostas-integral*:
Hadjicostas-integral holomorphic-on $\{z.\ Re\ z > -2\}$

proof −
  have $(\lambda z.\ (LBINT\ u{=}0..exp(-1).\ f\ z\ u) + (LBINT\ u{=}exp(-1)..1.\ f\ z\ u))$
      holomorphic-on $\{z.\ Re\ z > -2\}$
    by (*intro holomorphic-intros holo1 holo2*)
  also have *?this* $\longleftrightarrow$ $(\lambda z.\ LBINT\ u{=}0..1.\ f\ z\ u)$ *holomorphic-on* $\{z.\ Re\ z > -2\}$
    using *Hadjicostas-integral-integrable*
    by (*intro holomorphic-cong interval-integral-sum*)
      (*simp-all add*: *zero-ereal-def one-ereal-def min-def max-def*)
  also have $(\lambda z.\ LBINT\ u{=}0..1.\ f\ z\ u)$ = *Hadjicostas-integral*
    by (*simp add*: *Hadjicostas-integral-def*[*abs-def*] *f-def*)
  finally show *?thesis* **.**
qed

lemma *analytic-Hadjicostas-integral*:
  Hadjicostas-integral analytic-on $\{z.\ Re\ z > -2\}$
  by (*simp add*: *analytic-on-open open-halfspace-Re-gt holomorphic-Hadjicostas-integral*)

end

4.3 Analytic continuation and main result

Since we have already shown the formula for any real $z > -1$ and e. g. 0 is a limit point of that set, it extends to the full domain by analytic continuation. As a caveat, note that $\zeta(s)$ is *not* analytic at $z = 1$, so that we use an analytic continuation of $\zeta(z) - \frac{1}{z-1}$ to state the formula. This continuation is *pre-zeta 1*.

lemma *Hadjicostas-Chapman-formula-aux*:
  assumes *z*: $Re\ z > -2$
  shows *Hadjicostas-integral z* = *Gamma* $(z + 2) *$ *pre-zeta 1* $(z + 2)$
    (**is** - *z* = *?f z*)
proof (*rule analytic-continuation'*[*of Hadjicostas-integral*])
  show *Hadjicostas-integral holomorphic-on* $\{z.\ Re\ z > -2\}$
    by (*rule holomorphic-Hadjicostas-integral*)
  show *connected* $\{z.\ Re\ z > -2\}$
    by (*intro convex-connected convex-halfspace-Re-gt*)
  show *open* $\{z.\ Re\ z > -2\}$
    by (*auto simp*: *open-halfspace-Re-gt*)
  show $\{z.\ Re\ z > -1 \wedge Im\ z = 0\} \subseteq \{z.\ Re\ z > -2\}$ **and** $0 \in \{z.\ Re\ z > -2\}$
    by *auto*
  have $\forall n.\ 1\ /\ (of\text{-}nat\ (Suc\ n)) \in \{z.\ Re\ z > -1 \wedge Im\ z = 0\} - \{0\}$
    by (*auto simp*: *field-simps simp flip*: *of-nat-Suc*)
  moreover have $(\lambda n.\ 1\ /\ of\text{-}nat\ (Suc\ n) :: complex) \longlongrightarrow 0$
   by (*rule tendsto-divide-0*[*OF tendsto-const*] *filterlim-compose*[*OF tendsto-of-nat*]
filterlim-Suc)+
  ultimately show *0 islimpt* $\{z.\ Re\ z > -1 \wedge Im\ z = 0\}$
    unfolding *islimpt-sequential*
    by (*intro exI*[*of* - $\lambda n.\ 1\ /\ of\text{-}nat\ (Suc\ n) :: complex$]) *simp*
  show *?f holomorphic-on* $\{z.\ -\ 2 < Re\ z\}$

proof (*intro holomorphic-intros*)
fix *z* **assume** *z*: $z \in \{z.\ Re\ z > -2\}$
hence $z + 2 \notin \mathbb{R}_{\leq 0}$ **by** (*auto elim*!: *nonpos-Reals-cases simp*: *complex-eq-iff*)
thus $z + 2 \notin \mathbb{Z}_{\leq 0}$ **using** *nonpos-Ints-subset-nonpos-Reals* **by** *blast*
qed *auto*
next
fix *s* **assume** *s*: $s \in \{z.\ - 1 < Re\ z \wedge Im\ z = 0\}$
hence $s + 2 \neq 1$ **by** (*simp add*: *algebra-simps complex-eq-iff*)
have *ineq*: $x - ln\ x \geq 1$ **if** $x \in \{0<..<1\}$ **for** *x* :: *real*
using *ln-le-minus-one*[*of x*] *that* **by** (*simp add*: *algebra-simps*)
define *x* **where** $x = Re\ s$
from *s* **have** *x*: $x > -1$ **and** [*simp*]: $s = of\text{-}real\ x$
by (*auto simp*: *x-def complex-eq-iff*)
have *Hadjicostas-integral s* = (*LBINT u=0..1. of-real* ((−*ln u*) *powr x* / (1−*u*) ∗ (−*ln u* + *u* − 1)))
unfolding *Hadjicostas-integral-def*
by (*intro interval-lebesgue-integral-cong*) (*auto simp*: *einterval-def powr-Reals-eq*)
also have ... = *of-real* (*LBINT u=0..1.* (−*ln u*) *powr x* / (1−*u*) ∗ (−*ln u* + *u* − 1))
by (*subst interval-lebesgue-integral-of-real*) *auto*
also have (*LBINT u=0..1.* (− *ln u*) *powr x* / (1 − *u*) ∗ (− *ln u* + *u* − 1)) =
(∫ *u.* (− *ln u*) *powr x* / (1−*u*) ∗ (− *ln u* + *u* − 1) ∗ *indicator* {0<..<1} *u* ∂*lborel*)
by (*simp add*: *interval-integral-Ioo zero-ereal-def one-ereal-def set-lebesgue-integral-def mult-ac*)
also have ... = *enn2real* (*Hadjicostas-nn-integral x*)
unfolding *Hadjicostas-nn-integral-def* **using** *ineq*
by (*subst integral-eq-nn-integral*)
(*auto intro*!: *AE-I2 divide-nonneg-nonneg mult-nonneg-nonneg arg-cong*[**where** *f* = *enn2real*]
nn-integral-cong simp: *indicator-def*)
also have ... = *enn2real* (*ennreal* (*Gamma* (*x* + 2) ∗ (*Re* (*zeta* (*x* + 2)) − 1 / (*x* + 1))))
using *x* **by** (*subst Hadjicostas-Chapman-formula-real*) *auto*
also have ... = *Gamma* (*x* + 2) ∗ (*Re* (*zeta* (*x* + 2)) − 1 / (*x* + 1))
using *x real-zeta-ge-one-over-minus-one*[*of x* + 2]
by (*intro enn2real-ennreal mult-nonneg-nonneg Gamma-real-nonneg*) (*auto simp*: *add-ac*)
also have *complex-of-real* ... = *Gamma* (*s* + 2) ∗ (*zeta* (*s* + 2) − 1 / (*s* + 1))
using *x Gamma-complex-of-real*[*of x* + 2] **by** (*simp add*: *zeta-real'*)
also have (*zeta* (*s* + 2) − 1 / (*s* + 1)) = *pre-zeta* 1 (*s* + 2)
using ‹$s + 2 \neq 1$› **by** (*subst zeta-minus-pole-eq* [*symmetric*]) (*auto simp flip*: *of-nat-Suc*)
finally show *Hadjicostas-integral s* = *Gamma* (*s* + 2) ∗ *pre-zeta* 1 (*s* + 2) **.**
qed (*use assms* **in** *auto*)

The following form and the corollary are perhaps a bit nicer to read.

theorem *Hadjicostas-Chapman-formula*:

assumes *z*: *Re z* > −*2 z* ≠ −*1*
shows *Hadjicostas-integral z* = *Gamma* (*z* + *2*) ∗ (*zeta* (*z* + *2*) − *1* / (*z* + *1*))
proof −
from *z* **have** *z* + *1* ≠ *0*
by (*auto simp*: *complex-eq-iff*)
thus *?thesis* **using** *Hadjicostas-Chapman-formula-aux*[*of z*] *assms*
by (*subst* (*asm*) *zeta-minus-pole-eq* [*symmetric*]) (*auto simp*: *add-ac*)
qed

corollary *euler-mascheroni-integral-form*:
Hadjicostas-integral (−*1*) = *euler-mascheroni*
using *Hadjicostas-Chapman-formula-aux*[*of* −*1*] **by** *simp*

end

References

[1] T. M. Apostol. *Introduction to Analytic Number Theory*. Undergraduate Texts in Mathematics. Springer-Verlag, 1976.

[2] W. E. Briggs and S. Chowla. The power series coefficients of $\zeta(s)$. *The American Mathematical Monthly*, 62(5):323–325, 1955.

[3] R. Chapman. A proof of Hadjicostas's conjecture, 2004.

[4] P. Hadjicostas. A conjecture-generalization of Sondow's formula, 2004.

[5] J. Harrison. Formalizing an analytic proof of the Prime Number Theorem (dedicated to Mike Gordon on the occasion of his 60th birthday). *Journal of Automated Reasoning*, 43:243–261, 2009.

[6] D. Newman. *Analytic Number Theory*. Number 177 in Graduate Texts in Mathematics. Springer, 1998.

[7] J. Sondow. Double integrals for Euler's constant and $\ln \frac{4}{\pi}$, 2002.