Stirling's formula

Manuel Eberl

October 10, 2017

Abstract

This work contains a proof of Stirling's formula both for the factorial $n! \sim \sqrt{2\pi n}(n/e)^n$ on natural numbers and the real Gamma function $\Gamma(x) \sim \sqrt{2\pi/x}(x/e)^x$. The proof is based on work by Graham Jameson [3].

This is then used to derive the complete asymptotic expansion of the logarithmic Gamma function and its derivatives (the Polygamma functions) in terms of Bernoulli numbers.

Contents

1 Stirling's Formula

theory *Stirling-Formula*
imports
HOL−Analysis.Analysis
Landau-Symbols.Landau-Symbols
begin

context
begin

First, we define the S_n^* from Jameson's article:

private definition S' :: *nat* ⇒ *real* ⇒ *real* **where**
$S'\ n\ x = 1/(2*x) + (\sum r{=}1..{<}n.\ 1/(\mathit{of\text{-}nat}\ r{+}x)) + 1\ /(2*(n + x))$

Next, the trapezium (also called T in Jameson's article):

private definition T :: *real* ⇒ *real* **where**
$T\ x = 1/(2*x) + 1/(2*(x{+}1))$

Now we define The difference $\Delta(x)$:

```
private definition D :: real ⇒ real where
  D x = T x − ln (x + 1) + ln x

private lemma S′-telescope-trapezium:
  assumes n > 0
  shows   S′ n x = (∑ r<n. T (of-nat r + x))
proof (cases n)
  case (Suc m)
  hence m: Suc m = n by simp
  have (∑ r<n. T (of-nat r + x)) =
          (∑ r<Suc m. 1 / (2 ∗ real r + 2 ∗ x)) + (∑ r<n. 1 / (2 ∗ real (Suc r)
+ 2 ∗ x))
    unfolding m by (simp add: T-def sum.distrib algebra-simps)
  also have (∑ r<Suc m. 1 / (2 ∗ real r + 2 ∗ x)) =
              1/(2∗x) + (∑ r<m. 1 / (2 ∗ real (Suc r) + 2 ∗ x)) (is - = ?a + ?S)
    by (subst sum-lessThan-Suc-shift) simp
  also have (∑ r<n. 1 / (2 ∗ real (Suc r) + 2 ∗ x)) =
               ?S + 1 / (2∗(real m + x + 1)) (is - = - + ?b) by (simp add: Suc)
  also have ?a + ?S + (?S + ?b) = 2∗?S + ?a + ?b by (simp add: add-ac)
  also have 2 ∗ ?S = (∑ r=0..<m. 1 / (real (Suc r) + x))
    unfolding sum-distrib-left by (intro sum.cong) (auto simp add: divide-simps)
  also have (∑ r=0..<m. 1 / (real (Suc r) + x)) = (∑ r=Suc 0..<Suc m. 1 /
(real r + x))
    by (subst sum.atLeast-Suc-lessThan-Suc-shift) simp-all
  also have ... = (∑ r=1..<n. 1 / (real r + x)) unfolding m by simp
  also have ... + ?a + ?b = S′ n x by (simp add: S′-def Suc)
  finally show ?thesis ..
qed (insert assms, simp-all)

private lemma stirling-trapezium:
  assumes x: (x::real) > 0
  shows   D x ∈ {0 .. 1/(12∗x^2) − 1/(12 ∗ (x+1)^2)}
proof −
  define y where y = 1 / (2∗x + 1)
  from x have y: y > 0 y < 1 by (simp-all add: divide-simps y-def)

  from x have D x = T x − ln ((x + 1) / x) by (subst ln-div) (simp-all add:
D-def)
  also from x have (x + 1) / x = 1 + 1 / x by (simp add: field-simps)
  finally have D: D x = T x − ln (1 + 1/x) .

  from y have (λn. y ∗ y^n) sums (y ∗ (1 / (1 − y)))
    by (intro geometric-sums sums-mult) simp-all
  hence (λn. y ^ Suc n) sums (y / (1 − y)) by simp
  also from x have y / (1 − y) = 1 / (2∗x) by (simp add: y-def divide-simps)
  finally have ∗: (λn. y ^ Suc n) sums (1 / (2∗x)) .

  from y have (λn. (−y) ∗ (−y)^n) sums ((−y) ∗ (1 / (1 − (−y))))
```

```
    by (intro geometric-sums sums-mult) simp-all
  hence (λn. (−y) ^ Suc n) sums (−(y / (1 + y))) by simp
  also from x have y / (1 + y) = 1 / (2∗(x+1)) by (simp add: y-def divide-simps)
  finally have ∗∗: (λn. (−y) ^ Suc n) sums (−(1 / (2∗(x+1)))) .

  from sums-diff[OF ∗ ∗∗] have sum1: (λn. y ^ Suc n − (−y) ^ Suc n) sums T x
    by (simp add: T-def)

  from y have abs y < 1 abs (−y) < 1 by simp-all
  from sums-diff[OF this[THEN ln-series']]
    have (λn. y ^ n / real n − (−y) ^ n / real n) sums (ln (1 + y) − ln (1 − y)) by simp
  also from y have ln (1 + y) − ln (1 − y) = ln ((1 + y) / (1 − y)) by (simp add: ln-div)
  also from x have (1 + y) / (1 − y) = 1 + 1/x by (simp add: divide-simps y-def)
  finally have (λn. y^n / real n − (−y)^n / real n) sums ln (1 + 1/x) .
  hence sum2: (λn. y^Suc n / real (Suc n) − (−y)^Suc n / real (Suc n)) sums ln (1 + 1/x)
    by (subst sums-Suc-iff) simp

  from sum2 sum1 have ln (1 + 1/x) ≤ T x
  proof (rule sums-le [OF allI, rotated])
    fix n :: nat
    show y ^ Suc n / real (Suc n) − (−y) ^ Suc n / real (Suc n) ≤ y^Suc n − (−y) ^ Suc n
    proof (cases even n)
      case True
      hence eq: A − (−y) ^ Suc n / B = A + y ^ Suc n / B A − (−y) ^ Suc n = A + y ^ Suc n
        for A B by simp-all
      from y show ?thesis unfolding eq
        by (intro add-mono) (auto simp: divide-simps)
    qed simp-all
  qed
  hence D x ≥ 0 by (simp add: D)

  define c where c = (λn. if even n then 2 ∗ (1 − 1 / real (Suc n)) else 0)
  note sums-diff[OF sum1 sum2]
  also have (λn. y ^ Suc n − (−y) ^ Suc n − (y ^ Suc n / real (Suc n) −
                (−y) ^ Suc n / real (Suc n))) = (λn. c n ∗ y ^ Suc n)
    by (intro ext) (simp add: c-def algebra-simps)
  finally have sum3: (λn. c n ∗ y ^ Suc n) sums D x by (simp add: D)

  from y have (λn. y^2 ∗ (of-nat (Suc n) ∗ y^n)) sums (y^2 ∗ (1 / (1 − y)^2))
    by (intro sums-mult geometric-deriv-sums) simp-all
  hence (λn. of-nat (Suc n) ∗ y^(n+2)) sums (y^2 / (1 − y)^2)
    by (simp add: algebra-simps power2-eq-square)
  also from x have y^2 / (1 − y)^2 = 1 / (4∗x^2) by (simp add: y-def
```

divide-simps)
finally have ∗: (*λn. real* (*Suc n*) ∗ *y* ˆ (*Suc* (*Suc n*))) *sums* (*1* / (*4* ∗ x^2)) **by** *simp*

from *y* **have** (*λn. y*ˆ*2* ∗ (*of-nat* (*Suc n*) ∗ (−*y*)ˆ*n*)) *sums* (*y*ˆ*2* ∗ (*1* / (*1* − −*y*)ˆ*2*))
by (*intro sums-mult geometric-deriv-sums*) *simp-all*
hence (*λn. of-nat* (*Suc n*) ∗ (−*y*)ˆ(*n*+*2*)) *sums* (*y*ˆ*2* / (*1* + *y*)ˆ*2*)
by (*simp add*: *algebra-simps power2-eq-square*)
also from *x* **have** *y*ˆ*2* / (*1* + *y*)ˆ*2* = *1* / (*2*ˆ*2*∗(*x*+*1*)ˆ*2*)
unfolding *power-mult-distrib* [*symmetric*] **by** (*simp add*: *y-def divide-simps add-ac*)
finally have ∗∗: (*λn. real* (*Suc n*) ∗ (− *y*) ˆ (*Suc* (*Suc n*))) *sums* (*1* / (*4* ∗ (*x* + *1*)^2)) **by** *simp*

define *d* **where** *d* = (*λn. if even n then 2* ∗ *real n else 0*)
note *sums-diff*[*OF* ∗ ∗∗]
also have (*λn. real* (*Suc n*) ∗ *y*ˆ(*Suc* (*Suc n*)) − *real* (*Suc n*) ∗ (−*y*)ˆ(*Suc* (*Suc n*))) =
(*λn. d* (*Suc n*) ∗ *y* ˆ *Suc* (*Suc n*))
by (*intro ext*) (*simp-all add*: *d-def*)
finally have (*λn. d n* ∗ *y* ˆ *Suc n*) *sums* (*1* / (*4* ∗ x^2) − *1* / (*4* ∗ (*x* + *1*)^2))
by (*subst* (*asm*) *sums-Suc-iff*) (*simp add*: *d-def*)
from *sums-mult*[*OF this*, *of 1/3*] *x*
have *sum4*: (*λn. d n* / *3* ∗ *y* ˆ *Suc n*) *sums* (*1* / (*12* ∗ *x*ˆ*2*) − *1* / (*12* ∗ (*x* + *1*)ˆ*2*))
by (*simp add*: *field-simps*)

have *D x* ≤ (*1* / (*12* ∗ *x*ˆ*2*) − *1* / (*12* ∗ (*x* + *1*)ˆ*2*))
proof (*intro sums-le* [*OF - sum3 sum4*] *allI*)
fix *n* :: *nat*
define *c′* :: *nat* ⇒ *real*
where *c′* = (*λn. if odd n* ∨ *n* = *0 then 0 else if n* = *2 then 4/3 else 2*)
show *c n* ∗ *y* ˆ *Suc n* ≤ *d n* / *3* ∗ *y* ˆ *Suc n*
proof (*intro mult-right-mono*)
have *c n* ≤ *c′ n* **by** (*simp add*: *c-def c′-def*)
also consider *n* = *0* | *n* = *1* | *n* = *2* | *n* ≥ *3* **by** *force*
hence *c′ n* ≤ *d n* / *3* **by** *cases* (*simp-all add*: *c′-def d-def*)
finally show *c n* ≤ *d n* / *3* **.**
qed (*insert y*, *simp*)
qed

with ‹*D x* ≥ *0*› **show** *?thesis* **by** *simp*
qed

The following functions correspond to the $p_n(x)$ from the article. The special case $n = 0$ would not, strictly speaking, be necessary, but it allows some theorems to work even without the precondition $n \neq 0$:

private definition *p* :: *nat* ⇒ *real* ⇒ *real* **where**

$p\ n\ x = (if\ n = 0\ then\ 1/x\ else\ (\sum r{<}n.\ D\ (real\ r + x)))$

We can write the Digamma function in terms of S':

private lemma *S′-LIMSEQ-Digamma*:
assumes x: $x \neq 0$
shows $(\lambda n.\ ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)) \longrightarrow Digamma\ x$
proof −
define c **where** $c = (\lambda n.\ ln\ (real\ n) - (\sum r{<}n.\ inverse\ (x + real\ r)))$
have *eventually* $(\lambda n.\ 1\ /\ (2 * (x + real\ n)) = c\ n - (ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)))$ *at-top*
using *eventually-gt-at-top*[*of 0::nat*]
proof *eventually-elim*
fix n :: *nat*
assume n: $n > 0$
have $c\ n - (ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)) =$
$-(\sum r{<}n.\ inverse\ (real\ r + x)) + (1/x + (\sum r{=}1..{<}n.\ inverse\ (real\ r + x))) + 1/(2{*}(real\ n + x))$
using x **by** (*simp add*: *S′-def c-def field-simps*)
also have $1/x + (\sum r{=}1..{<}n.\ inverse\ (real\ r + x)) = (\sum r{<}n.\ inverse\ (real\ r + x))$
unfolding *lessThan-atLeast0* **using** n
by (*subst* (*2*) *sum-head-upt-Suc*) (*simp-all add*: *field-simps*)
finally show $1\ /\ (2 * (x + real\ n)) = c\ n - (ln\ (real\ n) - S'\ n\ x - 1/(2{*}x))$ **by** *simp*
qed
moreover have $(\lambda n.\ 1\ /\ (2 * (x + real\ n))) \longrightarrow 0$
by (*rule real-tendsto-divide-at-top tendsto-const filterlim-tendsto-pos-mult-at-top filterlim-tendsto-add-at-top filterlim-real-sequentially* | *simp*)+
ultimately have $(\lambda n.\ c\ n - (ln\ (real\ n) - S'\ n\ x - 1/(2{*}x))) \longrightarrow 0$
by (*rule Lim-transform-eventually*)
from *tendsto-minus*[*OF this*] **have** $(\lambda n.\ (ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)) - c\ n) \longrightarrow 0$ **by** *simp*
moreover from *Digamma-LIMSEQ*[*OF x*] **have** $c \longrightarrow Digamma\ x$ **by** (*simp add*: *c-def*)
ultimately show $(\lambda n.\ ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)) \longrightarrow Digamma\ x$
by (*rule Lim-transform* [*rotated*])
qed

Moreover, we can give an expansion of S' with the p as variation terms.

private lemma *S′-approx*:
$S'\ n\ x = ln\ (real\ n + x) - ln\ x + p\ n\ x$
proof (*cases* $n = 0$)
case *True*
thus *?thesis* **by** (*simp add*: *p-def S′-def*)
next
case *False*
hence $S'\ n\ x = (\sum r{<}n.\ T\ (real\ r + x))$
by (*subst S′-telescope-trapezium*) *simp-all*
also have $\ldots = (\sum r{<}n.\ ln\ (real\ r + x + 1) - ln\ (real\ r + x) + D\ (real\ r +$

x))
  by (*simp add*: *D-def*)
 also have ... = ($\sum$ *r*<*n*. *ln* (*real* (*Suc r*) + *x*) − *ln* (*real r* + *x*)) + *p n x*
  using *False* **by** (*simp add*: *sum.distrib add-ac p-def*)
 also have ($\sum$ *r*<*n*. *ln* (*real* (*Suc r*) + *x*) − *ln* (*real r* + *x*)) = *ln* (*real n* + *x*) − *ln x*
  by (*subst sum-lessThan-telescope*) *simp-all*
 finally show *?thesis* **.**
qed

We define the limit of the p (simply called $p(x)$ in Jameson's article):

private definition *P* :: *real* ⇒ *real* **where**
 P x = ($\sum$ *n*. *D* (*real n* + *x*))

private lemma *D-summable*:
 assumes *x*: *x* > *0*
 shows *summable* (*λn*. *D* (*real n* + *x*))
proof −
 have ∗: *summable* (*λn*. *1* / (*12* ∗ (*x* + *real n*)^2) − *1* / (*12* ∗ (*x* + *real* (*Suc n*))^2))
  by (*rule telescope-summable′ real-tendsto-divide-at-top tendsto-const filterlim-tendsto-pos-mult-at-top filterlim-pow-at-top filterlim-tendsto-add-at-top filterlim-real-sequentially* | *simp*)+
 show *summable* (*λn*. *D* (*real n* + *x*))
 proof (*rule summable-comparison-test*[*OF* - ∗], *rule exI*[*of* - *2*], *safe*)
  fix *n* :: *nat* **assume** *n* ≥ *2*
  show *norm* (*D* (*real n* + *x*)) ≤ *1* / (*12* ∗ (*x* + *real n*)ˆ*2*) − *1* / (*12* ∗ (*x* + *real* (*Suc n*))ˆ*2*)
   using *stirling-trapezium*[*of real n* + *x*] *x* **by** (*auto simp*: *algebra-simps*)
 qed
qed

private lemma *p-LIMSEQ*:
 assumes *x*: *x* > *0*
 shows (*λn*. *p n x*) ⟶ *P x*
proof −
 from *D-summable*[*OF x*] **have** (*λn*. *D* (*real n* + *x*)) *sums P x* **unfolding** *P-def*
  by (*simp add*: *sums-iff*)
 hence (*λn*. $\sum$ *r*<*n*. *D* (*real r* + *x*)) ⟶ *P x* **by** (*simp add*: *sums-def*)
 moreover from *eventually-gt-at-top*[*of 1*]
  have *eventually* (*λn*. ($\sum$ *r*<*n*. *D* (*real r* + *x*)) = *p n x*) *at-top*
  by *eventually-elim* (*auto simp*: *p-def*)
 ultimately show *?thesis* **by** (*rule Lim-transform-eventually* [*rotated*])
qed

This gives us an expansion of the Digamma function:

lemma *Digamma-approx*:
 assumes *x*: (*x* :: *real*) > *0*
 shows *Digamma x* = *ln x* − *1* / (*2* ∗ *x*) − *P x*

proof −
have *eventually* $(\lambda n.\ ln\ (inverse\ (1 + x\ /\ real\ n)) + ln\ x - 1/(2{*}x) - p\ n\ x =$
$ln\ (real\ n) - S'\ n\ x - 1/(2{*}x))$ *at-top*
using *eventually-gt-at-top*[*of* 1::*nat*]
proof *eventually-elim*
fix n :: *nat* **assume** n: $n > 1$
have $ln\ (real\ n) - S'\ n\ x = ln\ ((real\ n)\ /\ (real\ n + x)) + ln\ x - p\ n\ x$
using *assms n* **unfolding** S'*-approx* **by** (*subst ln-div*) (*auto simp*: *algebra-simps*)
also from n **have** $real\ n\ /\ (real\ n + x) = inverse\ (1 + x\ /\ real\ n)$ **by** (*simp add*: *field-simps*)
finally show $ln\ (inverse\ (1 + x\ /\ real\ n)) + ln\ x - 1/(2{*}x) - p\ n\ x =$
$ln\ (real\ n) - S'\ n\ x - 1/(2{*}x)$ **by** *simp*
qed
moreover have $(\lambda n.\ ln\ (inverse\ (1 + x\ /\ real\ n)) + ln\ x - 1/(2{*}x) - p\ n\ x)$
$\longrightarrow ln\ (inverse\ (1 + 0)) + ln\ x - 1/(2{*}x) - P\ x$
by (*rule tendsto-intros p-LIMSEQ x real-tendsto-divide-at-top*
filterlim-real-sequentially | *simp*)+
hence $(\lambda n.\ ln\ (inverse\ (1 + x\ /\ real\ n)) + ln\ x - 1/(2{*}x) - p\ n\ x)$
$\longrightarrow ln\ x - 1/(2{*}x) - P\ x$ **by** *simp*
ultimately have $(\lambda n.\ ln\ (real\ n) - S'\ n\ x - 1\ /\ (2 * x)) \longrightarrow ln\ x - 1/(2{*}x) - P\ x$
by (*rule Lim-transform-eventually*)
moreover from x **have** $(\lambda n.\ ln\ (real\ n) - S'\ n\ x - 1\ /\ (2 * x)) \longrightarrow Digamma\ x$
by (*intro* S'*-LIMSEQ-Digamma*) *simp-all*
ultimately show $Digamma\ x = ln\ x - 1\ /\ (2 * x) - P\ x$
by (*rule LIMSEQ-unique* [*rotated*])
qed

Next, we derive some bounds on P:

private lemma *p-ge-0*: $x > 0 \Longrightarrow p\ n\ x \geq 0$
using *stirling-trapezium*[*of real n + x* **for** *n*]
by (*auto simp add*: *p-def intro*!: *sum-nonneg*)

private lemma *P-ge-0*: $x > 0 \Longrightarrow P\ x \geq 0$
by (*rule tendsto-lowerbound*[*OF p-LIMSEQ*])
(*insert p-ge-0*[*of x*], *simp-all*)

private lemma *p-upper-bound*:
assumes $x > 0$ $n > 0$
shows $p\ n\ x \leq 1/(12{*}x\hat{\ }2)$
proof −
from *assms* **have** $p\ n\ x = (\sum r{<}n.\ D\ (real\ r + x))$
by (*simp add*: *p-def*)
also have $\ldots \leq (\sum r{<}n.\ 1/(12{*}(real\ r + x)\hat{\ }2) - 1/(12 * (real\ (Suc\ r) + x)\hat{\ }2))$
using *stirling-trapezium*[*of real r + x* **for** *r*] *assms*
by (*intro sum-mono*) (*simp add*: *add-ac*)
also have $\ldots = 1\ /\ (12 * x^2) - 1\ /\ (12 * (real\ n + x)^2)$

by (*subst sum-lessThan-telescope'*) *simp*
also have ... $\leq$ *1* / (*12* $*$ *x*ˆ*2*) **by** *simp*
finally show *?thesis* **.**
qed

private lemma *P-upper-bound*:
assumes *x* > *0*
shows *P x* $\leq$ *1*/(*12*$*$*x*ˆ*2*)
proof (*rule tendsto-upperbound*)
show *eventually* (λ*n. p n x* $\leq$ *1* / (*12* $*$ *x*ˆ*2*)) *at-top*
using *eventually-gt-at-top*[*of 0*]
by *eventually-elim* (*use p-upper-bound*[*of x*] *assms* **in** *auto*)
show (λ*n. p n x*) $\longrightarrow$ *P x*
by (*simp add*: *assms p-LIMSEQ*)
qed *auto*

We can now use this approximation of the Digamma function to approximate *ln-Gamma* (since the former is the derivative of the latter). We therefore define the function g from Jameson's article, which measures the difference between *ln-Gamma* and its approximation:

private definition *g* :: *real* $\Rightarrow$ *real* **where**
g x = *ln-Gamma x* − (*x* − *1*/*2*) $*$ *ln x* + *x*

private lemma *DERIV-g*: *x* > *0* $\Longrightarrow$ (*g has-field-derivative* −*P x*) (*at x*)
unfolding *g-def* [*abs-def*] **using** *Digamma-approx*[*of x*]
by (*auto intro*!: *derivative-eq-intros simp*: *field-simps*)

private lemma *isCont-P*:
assumes *x* > *0*
shows *isCont P x*
proof −
define *D'* :: *real* $\Rightarrow$ *real*
where *D'* = (λ*x.* − *1* / (*2* $*$ *x*ˆ*2* $*$ (*x*+*1*)ˆ*2*))
have *DERIV-D*: (*D has-field-derivative D' x*) (*at x*) **if** *x* > *0* **for** *x*
unfolding *D-def* [*abs-def*] *D'-def T-def*
by (*insert that*, (*rule derivative-eq-intros refl* | *simp*)+)
(*simp add*: *power2-eq-square divide-simps*, (*simp add*: *algebra-simps*)*?*)
note *this* [*THEN DERIV-chain2*, *derivative-intros*]

have (*P has-field-derivative* ($\sum$*n. D'* (*real n* + *x*))) (*at x*)
unfolding *P-def* [*abs-def*]
proof (*rule has-field-derivative-series'*)
show *convex* {*x*/*2*<..} **by** *simp*
next
fix *n* :: *nat* **and** *y* :: *real* **assume** *y*: *y* $\in$ {*x*/*2*<..}
with *assms* **have** *y* > *0* **by** *simp*
thus ((λ*a. D* (*real n* + *a*)) *has-real-derivative D'* (*real n* + *y*)) (*at y within* {*x*/*2*<..})
by (*auto intro*!: *derivative-eq-intros*)

```
  next
    from assms D-summable[of x] show summable (λn. D (real n + x)) by simp
  next
    show uniformly-convergent-on {x/2<..} (λn x. ∑i<n. D' (real i + x))
    proof (rule weierstrass-m-test')
      fix n :: nat and y :: real
      assume y: y ∈ {x/2<..}
      with assms have y > 0 by auto
      have norm (D' (real n + y)) = (1 / (2 ∗ (y + real n) ^2)) ∗ (1 / (y + real
(Suc n)) ^2)
        by (simp add: D'-def add-ac)
      also from y assms have ... ≤ (1 / (2 ∗ (x/2) ^2)) ∗ (1 / (real (Suc n)) ^2)
        by (intro mult-mono divide-left-mono power-mono) simp-all
      also have 1 / (real (Suc n)) ^2 = inverse ((real (Suc n)) ^2) by (simp add:
field-simps)
      finally show norm (D' (real n + y)) ≤ (1 / (2 ∗ (x/2) ^2)) ∗ inverse ((real
(Suc n)) ^2) .
    next
      show summable (λn. (1 / (2 ∗ (x/2) ^2)) ∗ inverse ((real (Suc n)) ^2))
        by (subst summable-Suc-iff, intro summable-mult inverse-power-summable)
simp-all
    qed
  qed (insert assms, simp-all add: interior-open)
  thus ?thesis by (rule DERIV-isCont)
qed

private lemma P-continuous-on [THEN continuous-on-subset]: continuous-on {0<..}
P
  by (intro continuous-at-imp-continuous-on ballI isCont-P) auto

private lemma P-integrable:
  assumes a: a > 0
  shows   P integrable-on {a..}
proof −
  define f where f = (λn x. if x ∈ {a..real n} then P x else 0)
  show P integrable-on {a..}
  proof (rule dominated-convergence)
    fix n :: nat
    from a have P integrable-on {a..real n}
      by (intro integrable-continuous-real P-continuous-on) auto
    hence f n integrable-on {a..real n}
      by (rule integrable-eq) (simp add: f-def)
    thus f n integrable-on {a..}
      by (rule integrable-on-superset) (auto simp: f-def)
  next
    fix n :: nat
    show ∀x∈{a..}. norm (f n x) ≤ of-real (1/12) ∗ (1 / x^2)
      using a P-ge-0 P-upper-bound by (auto simp: f-def)
  next
```

show (λx::*real*. *of-real* (*1/12*) $*$ (*1* / xˆ*2*)) *integrable-on* {*a*..}
using *has-integral-inverse-power-to-inf*[*of 2 a*] *a*
by (*intro integrable-on-cmult-left*) *auto*
next
show $\forall x \in \{a..\}$. (λn. *f n x*) $\longrightarrow$ *P x*
proof *safe*
fix *x* :: *real* **assume** *x*: $x \geq a$
have *eventually* (λn. *real n* $\geq$ *x*) *at-top*
using *filterlim-real-sequentially* **by** (*simp add*: *filterlim-at-top*)
with *x* **have** *eventually* (λn. *f n x* = *P x*) *at-top*
by (*auto elim*!: *eventually-mono simp*: *f-def*)
thus (λn. *f n x*) $\longrightarrow$ *P x* **by** (*simp add*: *Lim-eventually*)
qed
qed
qed

private definition *c* :: *real* **where** *c* = *integral* {*1*..} (λx. $-$*P x*) + *g 1*

We can now give bounds on g:

private lemma *g-bounds*:
assumes *x*: $x \geq 1$
shows *g x* $\in$ {*c*..*c* + *1*/(*12*$*$*x*)}
proof $-$
from *assms* **have** *int-nonneg*: *integral* {*x*..} *P* $\geq$ *0*
by (*intro Henstock-Kurzweil-Integration.integral-nonneg P-integrable*)
(*auto simp*: *P-ge-0*)
have *int-upper-bound*: *integral* {*x*..} *P* $\leq$ *1*/(*12*$*$*x*)
proof (*rule has-integral-le*)
from *x* **show** (*P has-integral integral* {*x*..} *P*) {*x*..}
by (*intro integrable-integral P-integrable*) *simp-all*
from *x has-integral-mult-right*[*OF has-integral-inverse-power-to-inf*[*of 2 x*], *of 1/12*]
show ((λx. *1*/(*12*$*$*x*ˆ*2*)) *has-integral* (*1*/(*12*$*$*x*))) {*x*..} **by** (*simp add*: *field-simps*)
qed (*insert P-upper-bound x*, *simp-all*)

note *DERIV-g* [*THEN DERIV-chain2*, *derivative-intros*]
from *assms* **have** *int1*: ((λx. $-$*P x*) *has-integral* (*g x* $-$ *g 1*)) {*1*..*x*}
by (*intro fundamental-theorem-of-calculus*)
(*auto simp*: *has-field-derivative-iff-has-vector-derivative* [*symmetric*]
intro!: *derivative-eq-intros*)
from *x* **have** *int2*: ((λx. $-$*P x*) *has-integral integral* {*x*..} (λx. $-$*P x*)) {*x*..}
by (*intro integrable-integral integrable-neg P-integrable*) *simp-all*
from *has-integral-Un*[*OF int1 int2*] *x*
have ((λx. $-$ *P x*) *has-integral g x* $-$ *g 1* $-$ *integral* {*x*..} *P*) ({*1*..*x*} $\cup$ {*x*..})
by (*simp add*: *max-def*)
also from *x* **have** {*1*..*x*} $\cup$ {*x*..} = {*1*..} **by** *auto*
finally have ((λx. $-$*P x*) *has-integral g x* $-$ *g 1* $-$ *integral* {*x*..} *P*) {*1*..} **.**
moreover have ((λx. $-$*P x*) *has-integral integral* {*1*..} (λx. $-$*P x*)) {*1*..}

by (*intro integrable-integral integrable-neg P-integrable*) *simp-all*
ultimately have $g\ x - g\ 1 - integral\ \{x..\}\ P = integral\ \{1..\}\ (\lambda x.\ -P\ x)$
by (*simp add*: *has-integral-unique*)
hence $g\ x = c + integral\ \{x..\}\ P$ **by** (*simp add*: *c-def algebra-simps*)
with *int-upper-bound int-nonneg* **show** $g\ x \in \{c..c + 1/(12*x)\}$ **by** *simp*
qed

Finally, we have bounds on *ln-Gamma*, *Gamma*, and *fact*.

private lemma *ln-Gamma-bounds-aux*:
$x \geq 1 \Longrightarrow ln\text{-}Gamma\ x \geq c + (x - 1/2) * ln\ x - x$
$x \geq 1 \Longrightarrow ln\text{-}Gamma\ x \leq c + (x - 1/2) * ln\ x - x + 1/(12*x)$
using *g-bounds*[*of x*] **by** (*simp-all add*: *g-def*)

private lemma *Gamma-bounds-aux*:
assumes *x*: $x \geq 1$
shows $Gamma\ x \geq exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x$
$Gamma\ x \leq exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x * exp\ (1/(12*x))$
proof −
have $exp\ (ln\text{-}Gamma\ x) \geq exp\ (c + (x - 1/2) * ln\ x - x)$
by (*subst exp-le-cancel-iff*, *rule ln-Gamma-bounds-aux*) (*simp add*: *x*)
with *x* **show** $Gamma\ x \geq exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x$
by (*simp add*: *Gamma-real-pos-exp exp-add exp-diff powr-def del*: *exp-le-cancel-iff*)
next
have $exp\ (ln\text{-}Gamma\ x) \leq exp\ (c + (x - 1/2) * ln\ x - x + 1/(12*x))$
by (*subst exp-le-cancel-iff*, *rule ln-Gamma-bounds-aux*) (*simp add*: *x*)
with *x* **show** $Gamma\ x \leq exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x * exp\ (1/(12*x))$
by (*simp add*: *Gamma-real-pos-exp exp-add exp-diff powr-def del*: *exp-le-cancel-iff*)
qed

private lemma *Gamma-asymp-equiv-aux*:
$Gamma \sim (\lambda x.\ exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x)$
proof (*rule asymp-equiv-sandwich*)
show *eventually* $(\lambda x.\ exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x \leq Gamma\ x)$ *at-top*
eventually $(\lambda x.\ exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x * exp\ (1/(12*x)) \geq Gamma\ x)$ *at-top*
using *eventually-ge-at-top*[*of 1*::*real*]
by (*eventually-elim*; *use Gamma-bounds-aux* **in** *force*)+
have $((\lambda x::real.\ exp\ (1\ /\ (12 * x))) \longrightarrow exp\ 0)$ *at-top*
by (*rule tendsto-intros real-tendsto-divide-at-top filterlim-tendsto-pos-mult-at-top*)+
(*simp-all add*: *filterlim-ident*)
hence $(\lambda x.\ exp\ (1\ /\ (12 * x))) \sim (\lambda x.\ 1)$
by (*intro asymp-equivI′*) *simp-all*
hence $(\lambda x.\ exp\ c * x\ powr\ (x - 1\ /\ 2)\ /\ exp\ x * 1) \sim$
$(\lambda x.\ exp\ c * x\ powr\ (x - 1\ /\ 2)\ /\ exp\ x * exp\ (1\ /\ (12 * x)))$
by (*intro asymp-equiv-mult asymp-equiv-refl*) (*simp add*: *asymp-equiv-sym*)
thus $(\lambda x.\ exp\ c * x\ powr\ (x - 1\ /\ 2)\ /\ exp\ x) \sim$
$(\lambda x.\ exp\ c * x\ powr\ (x - 1\ /\ 2)\ /\ exp\ x * exp\ (1\ /\ (12 * x)))$ **by** *simp*
qed *simp-all*

private lemma *exp-1-powr-real* [*simp*]: *exp* (*1*::*real*) *powr x* = *exp x*
  by (*simp add*: *powr-def*)

private lemma *fact-asymp-equiv-aux*:
  fact ∼ (λ*x*. *exp c* ∗ *sqrt* (*real x*) ∗ (*real x* / *exp 1*) *powr real x*)
proof −
  have *fact* ∼ (λ*n*. *Gamma* (*real* (*Suc n*))) **by** (*simp add*: *Gamma-fact*)
  also have *eventually* (λ*n*. *Gamma* (*real* (*Suc n*)) = *real n* ∗ *Gamma* (*real n*))
at-top
    using *eventually-gt-at-top*[*of 0*::*nat*]
    by *eventually-elim* (*insert Gamma-plus1*[*of real n* **for** *n*],
                    auto simp: *add-ac of-nat-in-nonpos-Ints-iff*)
  also have (λ*n*. *Gamma* (*real n*)) ∼ (λ*n*. *exp c* ∗ *real n powr* (*real n* − *1*/*2*) /
exp (*real n*))
   by (*rule asymp-equiv-compose*′[*OF Gamma-asymp-equiv-aux*] *filterlim-real-sequentially*)+
  also have *eventually* (λ*n*. *real n* ∗ (*exp c* ∗ *real n powr* (*real n* − *1* / *2*) / *exp*
(*real n*)) =
              exp c ∗ *sqrt* (*real n*) ∗ (*real n* / *exp 1*) *powr real n*) *at-top*
    using *eventually-gt-at-top*[*of 0*::*nat*]
  proof *eventually-elim*
    fix *n* :: *nat* **assume** *n*: *n* > *0*
    thus *real n* ∗ (*exp c* ∗ *real n powr* (*real n* − *1* / *2*) / *exp* (*real n*)) =
           exp c ∗ *sqrt* (*real n*) ∗ (*real n* / *exp 1*) *powr real n*
      by (*subst powr-diff*) (*simp-all add*: *powr-divide powr-half-sqrt field-simps*)
  qed
  finally show *?thesis* **by** − (*simp-all add*: *asymp-equiv-mult*)
qed

We still need to determine the constant term c, which we do using Wallis' product formula:

$$\prod_{n=1}^{\infty} \frac{4n^2}{4n^2 - 1} = \frac{\pi}{2}$$

private lemma *powr-mult-2*: (*x*::*real*) > *0* ⟹ *x powr* (*y* ∗ *2*) = (*x*ˆ*2*) *powr y*
 by (*subst mult.commute*, *subst powr-powr* [*symmetric*]) (*simp add*: *powr-numeral*)

private lemma *exp-mult-2*: *exp* (*y* ∗ *2* :: *real*) = *exp y* ∗ *exp y*
  by (*subst exp-add* [*symmetric*]) *simp*

private lemma *exp-c*: *exp c* = *sqrt* (*2*∗*pi*)
proof −
  define *p* **where** *p* = (λ*n*. ∏*k*=*1*..*n*. (*4*∗*real k*ˆ*2*) / (*4*∗*real k*ˆ*2* − *1*))

  have *p-0* [*simp*]: *p 0* = *1* **by** (*simp add*: *p-def*)
  have *p-Suc*: *p* (*Suc n*) = *p n* ∗ (*4* ∗ *real* (*Suc n*)ˆ*2*) / (*4* ∗ *real* (*Suc n*)ˆ*2* − *1*)
    for *n* **unfolding** *p-def* **by** (*subst prod-nat-ivl-Suc*′) *simp-all*
  have *p*: *p* = (λ*n*. *16* ˆ *n* ∗ *fact n* ˆ *4* / (*fact* (*2* ∗ *n*))^2 / (*2* ∗ *real n* + *1*))
  proof
    fix *n* :: *nat*

have $p\ n = (\prod k{=}1..n.\ (2{*}real\ k)\hat{\ }2\ /\ (2{*}real\ k\ -\ 1)\ /\ (2\ *\ real\ k\ +\ 1))$
unfolding *p-def* **by** (*intro prod.cong refl*) (*simp add*: *field-simps power2-eq-square*)
also have $\ldots\ = (\prod k{=}1..n.\ (2{*}real\ k)\hat{\ }2\ /\ (2{*}real\ k\ -\ 1))\ /\ (\prod k{=}1..n.\ (2\ *\ real\ (Suc\ k)\ -\ 1))$
by (*simp add*: *prod-dividef prod.distrib add-ac*)
also have $(\prod k{=}1..n.\ (2\ *\ real\ (Suc\ k)\ -\ 1)) = (\prod k{=}Suc\ 1..Suc\ n.\ (2\ *\ real\ k\ -\ 1))$
by (*subst prod.atLeast-Suc-atMost-Suc-shift*) *simp-all*
also have $\ldots\ = (\prod k{=}1..n.\ (2\ *\ real\ k\ -\ 1))\ *\ (2\ *\ real\ n\ +\ 1)$
by (*induction n*) (*simp-all add*: *prod-nat-ivl-Suc′*)
also have $(\prod k\ =\ 1..n.\ (2\ *\ real\ k)^2\ /\ (2\ *\ real\ k\ -\ 1))\ /\ \ldots\ =$
$(\prod k\ =\ 1..n.\ (2\ *\ real\ k)\hat{\ }2\ /\ (2\ *\ real\ k\ -\ 1)\hat{\ }2)\ /\ (2\ *\ real\ n\ +\ 1)$
unfolding *power2-eq-square* **by** (*simp add*: *prod.distrib prod-dividef*)
also have $(\prod k\ =\ 1..n.\ (2\ *\ real\ k)\hat{\ }2\ /\ (2\ *\ real\ k\ -\ 1)\hat{\ }2) =$
$(\prod k\ =\ 1..n.\ (2\ *\ real\ k)\hat{\ }4\ /\ ((2{*}real\ k){*}(2\ *\ real\ k\ -\ 1))\hat{\ }2)$
by (*intro prod.cong refl*)
(*simp add*: *divide-simps*, (*simp add*: *field-simps power2-eq-square eval-nat-numeral*))
also have $\ldots\ = 16\hat{\ }n\ *\ fact\ n\hat{\ }4\ /\ (\prod k{=}1..n.\ (2{*}real\ k)\ *\ (2{*}real\ k\ -\ 1))\hat{\ }2$
by (*simp add*: *prod.distrib prod-dividef fact-prod*
prod-power-distrib [*symmetric*] *prod-constant*)
also have $(\prod k{=}1..n.\ (2{*}real\ k)\ *\ (2{*}real\ k\ -\ 1)) = fact\ (2{*}n)$
by (*induction n*) (*simp-all add*: *algebra-simps prod-nat-ivl-Suc′*)
finally show $p\ n = 16\ \hat{\ }\ n\ *\ fact\ n\ \hat{\ }\ 4\ /\ (fact\ (2\ *\ n))^2\ /\ (2\ *\ real\ n\ +\ 1)$ **.**
qed

have $p \sim (\lambda n.\ 16\ \hat{\ }\ n\ *\ fact\ n\ \hat{\ }\ 4\ /\ (fact\ (2\ *\ n))^2\ /\ (2\ *\ real\ n\ +\ 1))$
by (*simp add*: *p*)
also have $\ldots\ \sim (\lambda n.\ 16\hat{\ }n\ *\ (exp\ c\ *\ sqrt\ (real\ n)\ *\ (real\ n\ /\ exp\ 1)\ powr\ real\ n)\hat{\ }4\ /$
$(exp\ c\ *\ sqrt\ (real\ (2{*}n))\ *\ (real\ (2{*}n)\ /\ exp\ 1)\ powr\ real\ (2{*}n))\hat{\ }2\ /$
$(2\ *\ real\ n\ +\ 1))$ (**is** - $\sim$ *?f*)
by (*intro asymp-equiv-mult asymp-equiv-divide asymp-equiv-refl mult-nat-left-at-top*
fact-asymp-equiv-aux asymp-equiv-power asymp-equiv-compose′[*OF fact-asymp-equiv-aux*])
simp-all
also have *eventually* $(\lambda n.\ \ldots\ n = exp\ c\ \hat{\ }\ 2\ /\ (4\ +\ 2/n))$ *at-top*
using *eventually-gt-at-top*[*of 0*::*nat*]
proof *eventually-elim*
fix n :: *nat* **assume** n: $n > 0$
have [*simp*]: $16\hat{\ }n = 4\hat{\ }n\ *\ (4\hat{\ }n\ ::\ real)$ **by** (*simp add*: *power-mult-distrib* [*symmetric*])
from n **have** $?f\ n = exp\ c\ \hat{\ }\ 2\ *\ (n\ /\ (2{*}(2{*}n{+}1)))$
by (*simp add*: *power-mult-distrib divide-simps powr-mult real-sqrt-power-even*)
(*simp add*: *field-simps power2-eq-square eval-nat-numeral powr-mult-2*
exp-mult-2 powr-realpow)
also from n **have** $\ldots\ = exp\ c\ \hat{\ }\ 2\ /\ (4\ +\ 2/n)$ **by** (*simp add*: *field-simps*)
finally show $?f\ n = \ldots$ **.**
qed

also have $(\lambda x.\ 4 + 2\ /\ real\ x) \sim (\lambda x.\ 4)$
  by (*subst asymp-equiv-add-right*) *auto*
finally have $p \longrightarrow exp\ c\ \hat{}\ 2\ /\ 4$
  by (*rule asymp-equivD-const*) (*simp-all add*: *asymp-equiv-divide*)
moreover have $p \longrightarrow pi\ /\ 2$ **unfolding** *p-def* **by** (*rule wallis*)
ultimately have $exp\ c\ \hat{}\ 2\ /\ 4 = pi\ /\ 2$ **by** (*rule LIMSEQ-unique*)
hence $2 * pi = exp\ c\ \hat{}\ 2$ **by** *simp*
also have $sqrt\ (exp\ c\ \hat{}\ 2) = exp\ c$ **by** *simp*
finally show $exp\ c = sqrt\ (2 * pi)$ **..**
qed

private lemma *c*: $c = ln\ (2{*}pi)\ /\ 2$
proof −
note *exp-c* [*symmetric*]
also have $ln\ (exp\ c) = c$ **by** *simp*
finally show *?thesis* **by** (*simp add*: *ln-sqrt*)
qed

This gives us the final bounds:

theorem *Gamma-bounds*:
assumes $x \geq 1$
shows $Gamma\ x \geq sqrt\ (2{*}pi/x) * (x\ /\ exp\ 1)\ powr\ x$ (**is** *?th1*)
$Gamma\ x \leq sqrt\ (2{*}pi/x) * (x\ /\ exp\ 1)\ powr\ x * exp\ (1\ /\ (12 * x))$ (**is** *?th2*)
proof −
from *assms* **have** $exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x = sqrt\ (2{*}pi/x) * (x\ /\ exp\ 1)\ powr\ x$
  by (*subst powr-diff*)
  (*simp add*: *exp-c real-sqrt-divide powr-divide powr-half-sqrt field-simps*)
with *Gamma-bounds-aux*[*OF assms*] **show** *?th1 ?th2* **by** *simp-all*
qed

theorem *ln-Gamma-bounds*:
assumes $x \geq 1$
shows $ln\text{-}Gamma\ x \geq ln\ (2{*}pi/x)\ /\ 2 + x * ln\ x - x$ (**is** *?th1*)
$ln\text{-}Gamma\ x \leq ln\ (2{*}pi/x)\ /\ 2 + x * ln\ x - x + 1/(12{*}x)$ (**is** *?th2*)
proof −
from *ln-Gamma-bounds-aux*[*OF assms*] *assms* **show** *?th1 ?th2*
  by (*simp-all add*: *c field-simps ln-div*)
from *assms* **have** $exp\ c * x\ powr\ (x - 1/2)\ /\ exp\ x = sqrt\ (2{*}pi/x) * (x\ /\ exp\ 1)\ powr\ x$
  by (*subst powr-diff*)
  (*simp add*: *exp-c real-sqrt-divide powr-divide powr-half-sqrt field-simps*)
qed

theorem *fact-bounds*:
assumes $n > 0$
shows $(fact\ n :: real) \geq sqrt\ (2{*}pi{*}n) * (n\ /\ exp\ 1)\ \hat{}\ n$ (**is** *?th1*)
$(fact\ n :: real) \leq sqrt\ (2{*}pi{*}n) * (n\ /\ exp\ 1)\ \hat{}\ n * exp\ (1\ /\ (12 * n))$ (**is**

?th2)
proof −
  from *assms* **have** *n*: *real n* ≥ *1* **by** *simp*
  from *assms Gamma-plus1*[*of real n*]
    have *real n* ∗ *Gamma* (*real n*) = *Gamma* (*real* (*Suc n*))
    by (*simp add*: *of-nat-in-nonpos-Ints-iff add-ac*)
  also have *Gamma* (*real* (*Suc n*)) = *fact n* **by** (*subst Gamma-fact* [*symmetric*]) *simp*
  finally have ∗: *fact n* = *real n* ∗ *Gamma* (*real n*) **by** *simp*

  have *2*∗*pi*/*n* = *2*∗*pi*∗*n* / *n*ˆ*2* **by** (*simp add*: *power2-eq-square*)
  also have *sqrt* ... = *sqrt* (*2*∗*pi*∗*n*) / *n* **by** (*subst real-sqrt-divide*) *simp-all*
  also have *real n* ∗ ... = *sqrt* (*2*∗*pi*∗*n*) **by** *simp*
  finally have ∗∗: *real n* ∗ *sqrt* (*2*∗*pi*/*real n*) = *sqrt* (*2*∗*pi*∗*real n*) **.**

  note ∗
  also note *Gamma-bounds*(*2*)[*OF n*]
  also have *real n* ∗ (*sqrt* (*2* ∗ *pi* / *real n*) ∗ (*real n* / *exp 1*) *powr real n* ∗
        exp (*1* / (*12* ∗ *real n*))) =
      (*real n* ∗ *sqrt* (*2*∗*pi*/*n*)) ∗ (*n* / *exp 1*) *powr n* ∗ *exp* (*1* / (*12* ∗ *n*))
    by (*simp add*: *algebra-simps*)
  also from *n* **have** (*real n* / *exp 1*) *powr real n* = (*real n* / *exp 1*) ˆ *n*
    by (*subst powr-realpow*) *simp-all*
  also note ∗∗
  finally show *?th2* **by** − (*insert assms*, *simp-all*)

  have *sqrt* (*2*∗*pi*∗*n*) ∗ (*n* / *exp 1*) *powr n* = *n* ∗ (*sqrt* (*2*∗*pi*/*n*) ∗ (*n* / *exp 1*) *powr n*)
    by (*subst* ∗∗ [*symmetric*]) (*simp add*: *field-simps*)
  also from *assms* **have** ... ≤ *real n* ∗ *Gamma* (*real n*)
    by (*intro mult-left-mono Gamma-bounds*(*1*)) *simp-all*
  also from *n* **have** (*real n* / *exp 1*) *powr real n* = (*real n* / *exp 1*) ˆ *n*
    by (*subst powr-realpow*) *simp-all*
  also note ∗ [*symmetric*]
  finally show *?th1* **.**
qed

theorem *ln-fact-bounds*:
  assumes *n* > *0*
  shows *ln* (*fact n* :: *real*) ≥ *ln* (*2*∗*pi*∗*n*)/*2* + *n* ∗ *ln n* − *n* (**is** *?th1*)
      ln (*fact n* :: *real*) ≤ *ln* (*2*∗*pi*∗*n*)/*2* + *n* ∗ *ln n* − *n* + *1*/(*12*∗*real n*) (**is** *?th2*)
proof −
  have *ln* (*fact n* :: *real*) ≥ *ln* (*sqrt* (*2*∗*pi*∗*real n*)∗(*real n*/*exp 1*)ˆ*n*)
    using *fact-bounds*(*1*)[*OF assms*] *assms* **by** (*subst ln-le-cancel-iff*) *auto*
  also from *assms* **have** *ln* (*sqrt* (*2*∗*pi*∗*real n*)∗(*real n*/*exp 1*)ˆ*n*) = *ln* (*2*∗*pi*∗*n*)/*2* + *n* ∗ *ln n* − *n*
    by (*simp add*: *ln-mult ln-div ln-realpow ln-sqrt field-simps*)
  finally show *?th1* **.**

```
next
  have ln (fact n :: real) ≤ ln (sqrt (2*pi*real n) * (real n/exp 1)^n * exp
(1/(12*real n)))
    using fact-bounds(2)[OF assms] assms by (subst ln-le-cancel-iff) auto
  also from assms have ... = ln (2*pi*n)/2 + n * ln n − n + 1/(12*real n)
    by (simp add: ln-mult ln-div ln-realpow ln-sqrt field-simps)
  finally show ?th2 .
qed

theorem Gamma-asymp-equiv:
  Gamma ∼ (λx. sqrt (2*pi/x) * (x / exp 1) powr x :: real)
proof −
  note Gamma-asymp-equiv-aux
  also have eventually (λx. exp c * x powr (x − 1/2) / exp x =
              sqrt (2*pi/x) * (x / exp 1) powr x) at-top
    using eventually-gt-at-top[of 0::real]
  proof eventually-elim
    fix x :: real assume x: x > 0
    thus exp c * x powr (x − 1/2) / exp x = sqrt (2*pi/x) * (x / exp 1) powr x
      by (subst powr-diff)
         (simp add: exp-c powr-half-sqrt powr-divide field-simps real-sqrt-divide)
  qed
  finally show ?thesis .
qed

theorem fact-asymp-equiv:
  fact ∼ (λn. sqrt (2*pi*n) * (n / exp 1) ^ n :: real)
proof −
  note fact-asymp-equiv-aux
  also have eventually (λn. exp c * sqrt (real n) = sqrt (2 * pi * real n)) at-top
   using eventually-gt-at-top[of 0::nat] by eventually-elim (simp add: exp-c real-sqrt-mult)
  also have eventually (λn. (n / exp 1) powr n = (n / exp 1) ^ n) at-top
   using eventually-gt-at-top[of 0::nat] by eventually-elim (simp add: powr-realpow)
  finally show ?thesis .
qed

end

end
```

2 Complete asymptotics of the logarithmic Gamma function

```
theory Ln-Gamma-Asymptotics
imports
  HOL−Analysis.Analysis
  Bernoulli.Bernoulli-FPS
  Bernoulli.Periodic-Bernpoly
```

Stirling-Formula
begin

2.1 Auxiliary Facts

lemma *filterlim-at-infinity-conv-norm-at-top*:
filterlim f at-infinity G $\longleftrightarrow$ *filterlim* (λx. *norm* (*f x*)) *at-top G*
by (*auto simp*: *filterlim-at-infinity*[*OF order.refl*] *filterlim-at-top-gt*[*of - - 0*])

corollary *Ln-times-of-nat*:
$[\![r > 0; z \neq 0]\!] \Longrightarrow$ *Ln*(*of-nat r* $*$ *z* :: *complex*) = *ln* (*of-nat r*) + *Ln*(*z*)
using *Ln-times-of-real*[*of of-nat r z*] **by** *simp*

lemma *tendsto-of-real-0-I*:
(*f* $\longrightarrow$ *0*) *G* $\Longrightarrow$ ((λx. (*of-real* (*f x*))) $\longrightarrow$ (*0* ::'*a*::*real-normed-div-algebra*)) *G*
by (*subst* (*asm*) *tendsto-of-real-iff* [*symmetric*]) *simp*

lemma *negligible-atLeastAtMostI*: *b* $\leq$ *a* $\Longrightarrow$ *negligible* {*a*..(*b*::*real*)}
by (*cases b* < *a*) *auto*

lemma *integrable-on-negligible*:
negligible A $\Longrightarrow$ (*f* :: '*n* :: *euclidean-space* $\Rightarrow$ '*a* :: *banach*) *integrable-on A*
by (*subst integrable-spike-set-eq*[*of -* {}]) (*simp-all add*: *integrable-on-empty*)

lemma *vector-derivative-cong-eq*:
assumes *eventually* (λx. *x* $\in$ *A* $\longrightarrow$ *f x* = *g x*) (*nhds x*) *x* = *y A* = *B x* $\in$ *A*
shows *vector-derivative f* (*at x within A*) = *vector-derivative g* (*at y within B*)
proof −
from *eventually-nhds-x-imp-x*[*OF assms*(*1*)] *assms*(*4*) **have** *f x* = *g x* **by** *blast*
hence (λD. (*f has-vector-derivative D*) (*at x within A*)) =
(λD. (*g has-vector-derivative D*) (*at x within A*)) **using** *assms*
by (*intro ext has-vector-derivative-cong-ev refl assms*) *simp-all*
thus *?thesis* **by** (*simp add*: *vector-derivative-def assms*)
qed

lemma *differentiable-of-real* [*simp*]: *of-real differentiable at x within A*
proof −
have (*of-real has-vector-derivative 1*) (*at x within A*)
by (*auto intro*!: *derivative-eq-intros*)
thus *?thesis* **by** (*rule differentiableI-vector*)
qed

lemma *higher-deriv-cong-ev*:
assumes *eventually* (λx. *f x* = *g x*) (*nhds x*) *x* = *y*
shows (*deriv* ^^ *n*) *f x* = (*deriv* ^^ *n*) *g y*
proof −
from *assms*(*1*) **have** *eventually* (λx. (*deriv* ^^ *n*) *f x* = (*deriv* ^^ *n*) *g x*) (*nhds x*)

proof (*induction n arbitrary*: *f g*)
case (*Suc n*)
from *Suc.prems* **have** *eventually* (λy. *eventually* (λz. *f z* = *g z*) (*nhds y*)) (*nhds x*)
by (*simp add*: *eventually-eventually*)
hence *eventually* (λx. *deriv f x* = *deriv g x*) (*nhds x*)
by *eventually-elim* (*rule deriv-cong-ev*, *simp-all*)
thus *?case* **by** (*auto intro*!: *deriv-cong-ev Suc simp*: *funpow-Suc-right simp del*: *funpow.simps*)
qed *auto*
from *eventually-nhds-x-imp-x*[*OF this*] *assms*(*2*) **show** *?thesis* **by** *simp*
qed

lemma *deriv-of-real* [*simp*]:
at x within A $\neq$ *bot* $\Longrightarrow$ *vector-derivative of-real* (*at x within A*) = *1*
by (*auto intro*!: *vector-derivative-within derivative-eq-intros*)

lemma *deriv-Re* [*simp*]: *deriv Re* = (λ-. *1*)
by (*auto intro*!: *DERIV-imp-deriv simp*: *fun-eq-iff*)

lemma *vector-derivative-of-real-left*:
assumes *f differentiable at x*
shows *vector-derivative* (λx. *of-real* (*f x*)) (*at x*) = *of-real* (*deriv f x*)
proof −
have *vector-derivative* (*of-real* $\circ$ *f*) (*at x*) = (*of-real* (*deriv f x*))
by (*subst vector-derivative-chain-at*)
(*simp-all add*: *scaleR-conv-of-real field-derivative-eq-vector-derivative assms*)
thus *?thesis* **by** (*simp add*: *o-def*)
qed

lemma *vector-derivative-of-real-right*:
assumes *f field-differentiable at* (*of-real x*)
shows *vector-derivative* (λx. *f* (*of-real x*)) (*at x*) = *deriv f* (*of-real x*)
proof −
have *vector-derivative* (*f* $\circ$ *of-real*) (*at x*) = *deriv f* (*of-real x*)
using *assms* **by** (*subst vector-derivative-chain-at-general*) *simp-all*
thus *?thesis* **by** (*simp add*: *o-def*)
qed

lemma *Ln-holomorphic* [*holomorphic-intros*]:
assumes $A \cap \mathbb{R}_{\leq 0} = \{\}$
shows *Ln holomorphic-on* (*A* :: *complex set*)
proof (*intro holomorphic-onI*)
fix *z* **assume** $z \in A$
with *assms* **have** (*Ln has-field-derivative inverse z*) (*at z within A*)
by (*auto intro*!: *derivative-eq-intros*)
thus *Ln field-differentiable at z within A* **by** (*auto simp*: *field-differentiable-def*)
qed

lemma *ln-Gamma-holomorphic* [*holomorphic-intros*]:
  assumes $A \cap \mathbb{R}_{\leq 0} = \{\}$
  shows *ln-Gamma holomorphic-on* (*A* :: *complex set*)
proof (*intro holomorphic-onI*)
  fix *z* **assume** $z \in A$
  with *assms* **have** (*ln-Gamma has-field-derivative Digamma z*) (*at z within A*)
    by (*auto intro*!: *derivative-eq-intros*)
  thus *ln-Gamma field-differentiable at z within A* **by** (*auto simp*: *field-differentiable-def*)
qed

lemma *higher-deriv-Polygamma*:
  assumes $z \notin \mathbb{Z}_{\leq 0}$
  shows (*deriv* ^^ *n*) (*Polygamma m*) *z* =
    Polygamma (*m* + *n*) (*z* :: *'a* :: {*real-normed-field*,*euclidean-space*})
proof −
  have *eventually* ($\lambda u.$ (*deriv* ^^ *n*) (*Polygamma m*) *u* = *Polygamma* (*m* + *n*) *u*)
(*nhds z*)
  proof (*induction n*)
    case (*Suc n*)
    from *Suc.IH* **have** *eventually* ($\lambda z.$ *eventually* ($\lambda u.$ (*deriv* ^^ *n*) (*Polygamma*
m) *u* = *Polygamma* (*m* + *n*) *u*) (*nhds z*)) (*nhds z*)
      by (*simp add*: *eventually-eventually*)
    hence *eventually* ($\lambda z.$ *deriv* ((*deriv* ^^ *n*) (*Polygamma m*)) *z* =
      deriv (*Polygamma* (*m* + *n*)) *z*) (*nhds z*)
      by *eventually-elim* (*intro deriv-cong-ev refl*)
    moreover have *eventually* ($\lambda z.$ $z \in$ *UNIV* $- \mathbb{Z}_{\leq 0}$) (*nhds z*) **using** *assms*
      by (*intro eventually-nhds-in-open open-Diff open-UNIV*) *auto*
    ultimately show *?case* **by** *eventually-elim* (*simp-all add*: *deriv-Polygamma*)
  qed *simp-all*
  thus *?thesis* **by** (*rule eventually-nhds-x-imp-x*)
qed

lemma *higher-deriv-cmult*:
  assumes *f holomorphic-on A* $x \in A$ *open A*
  shows (*deriv* ^^ *j*) ($\lambda x.$ *c* ∗ *f x*) *x* = *c* ∗ (*deriv* ^^ *j*) *f x*
  using *assms*
proof (*induction j arbitrary*: *f x*)
  case (*Suc j f x*)
  have *deriv* ((*deriv* ^^ *j*) ($\lambda x.$ *c* ∗ *f x*)) *x* = *deriv* ($\lambda x.$ *c* ∗ (*deriv* ^^ *j*) *f x*) *x*
    using *eventually-nhds-in-open*[*of A x*] *assms*(*2*,*3*) *Suc.prems*
    by (*intro deriv-cong-ev refl*) (*auto elim*!: *eventually-mono simp*: *Suc.IH*)
  also have . . . = *c* ∗ *deriv* ((*deriv* ^^ *j*) *f*) *x* **using** *Suc.prems assms*(*2*,*3*)
   by (*intro deriv-cmult holomorphic-on-imp-differentiable-at holomorphic-higher-deriv*)
auto
  finally show *?case* **by** *simp*
qed *simp-all*

lemma *higher-deriv-ln-Gamma-complex*:
  assumes (*x*::*complex*) $\notin \mathbb{R}_{\leq 0}$

shows (*deriv* ^^ *j*) *ln-Gamma x* = (*if j* = *0 then ln-Gamma x else Polygamma* (*j* − *1*) *x*)
proof (*cases j*)
  case (*Suc j'*)
  have (*deriv* ^^ *j'*) (*deriv ln-Gamma*) *x* = (*deriv* ^^ *j'*) *Digamma x*
    using *eventually-nhds-in-open*[*of UNIV* − $\mathbb{R}_{\leq 0}$ *x*] *assms*
    by (*intro higher-deriv-cong-ev refl*)
      (*auto elim*!: *eventually-mono simp*: *open-Diff deriv-ln-Gamma-complex*)
  also have ... = *Polygamma j' x* **using** *assms*
    by (*subst higher-deriv-Polygamma*)
      (*auto elim*!: *nonpos-Ints-cases simp*: *complex-nonpos-Reals-iff*)
 finally show *?thesis* **using** *Suc* **by** (*simp del*: *funpow.simps add*: *funpow-Suc-right*)
qed *simp-all*

lemma *higher-deriv-ln-Gamma-real*:
  assumes (*x*::*real*) > *0*
  shows (*deriv* ^^ *j*) *ln-Gamma x* = (*if j* = *0 then ln-Gamma x else Polygamma* (*j* − *1*) *x*)
proof (*cases j*)
  case (*Suc j'*)
  have (*deriv* ^^ *j'*) (*deriv ln-Gamma*) *x* = (*deriv* ^^ *j'*) *Digamma x*
    using *eventually-nhds-in-open*[*of* {*0*<..} *x*] *assms*
    by (*intro higher-deriv-cong-ev refl*)
      (*auto elim*!: *eventually-mono simp*: *open-Diff deriv-ln-Gamma-real*)
  also have ... = *Polygamma j' x* **using** *assms*
    by (*subst higher-deriv-Polygamma*)
      (*auto elim*!: *nonpos-Ints-cases simp*: *complex-nonpos-Reals-iff*)
 finally show *?thesis* **using** *Suc* **by** (*simp del*: *funpow.simps add*: *funpow-Suc-right*)
qed *simp-all*

lemma *higher-deriv-ln-Gamma-complex-of-real*:
  assumes (*x* :: *real*) > *0*
  shows (*deriv* ^^ *j*) *ln-Gamma* (*complex-of-real x*) = *of-real* ((*deriv* ^^ *j*) *ln-Gamma x*)
    using *assms*
    by (*auto simp*: *higher-deriv-ln-Gamma-real higher-deriv-ln-Gamma-complex*
          ln-Gamma-complex-of-real Polygamma-of-real)

lemma *stirling-limit-aux1*:
  ((λ*y*. *Ln* (*1* + *z* ∗ *of-real y*) / *of-real y*) ⟶ *z*) (*at-right 0*) **for** *z* :: *complex*
proof (*cases z* = *0*)
  case *True*
  then show *?thesis* **by** *simp*
next
  case *False*
  have ((λ*y*. *ln* (*1* + *z* ∗ *of-real y*)) *has-vector-derivative 1* ∗ *z*) (*at 0*)
    by (*rule has-vector-derivative-real-complex*) (*auto intro*!: *derivative-eq-intros*)
  then have (λ*y*. (*Ln* (*1* + *z* ∗ *of-real y*) − *of-real y* ∗ *z*) / *of-real* |*y*|) −*0*→ *0*

by (*auto simp add*: *has-vector-derivative-def has-derivative-def netlimit-at scaleR-conv-of-real field-simps*)
then have $((\lambda y.\ (Ln\ (1 + z * of\text{-}real\ y) - of\text{-}real\ y * z)\ /\ of\text{-}real\ |y|) \longrightarrow 0)$ $(at\text{-}right\ 0)$
by (*rule filterlim-mono*[*OF - - at-le*]) *simp-all*
also have *?this* $\longleftrightarrow$ $((\lambda y.\ Ln\ (1 + z * of\text{-}real\ y)\ /\ (of\text{-}real\ y) - z) \longrightarrow 0)$ $(at\text{-}right\ 0)$
using *eventually-at-right-less*[*of 0*::*real*]
by (*intro filterlim-cong refl*) (*auto elim*!: *eventually-mono simp*: *field-simps*)
finally show *?thesis* **by** (*simp only*: *LIM-zero-iff*)
qed

lemma *stirling-limit-aux2*:
$((\lambda y.\ y * Ln\ (1 + z\ /\ of\text{-}real\ y)) \longrightarrow z)$ *at-top* **for** z :: *complex*
using *stirling-limit-aux1*[*of z*] **by** (*subst filterlim-at-top-to-right*) (*simp add*: *field-simps*)

lemma *Union-atLeastAtMost*:
assumes $N > 0$
shows $(\bigcup n\in\{0..<N\}.\ \{real\ n..real\ (n + 1)\}) = \{0..real\ N\}$
proof (*intro equalityI subsetI*)
fix x **assume** x: $x \in \{0..real\ N\}$
thus $x \in (\bigcup n\in\{0..<N\}.\ \{real\ n..real\ (n + 1)\})$
proof (*cases* $x = real\ N$)
case *True*
with *assms* **show** *?thesis* **by** (*auto intro*!: *bexI*[*of - N − 1*])
next
case *False*
with x **have** x: $x \geq 0$ $x < real\ N$ **by** *simp-all*
hence $x \geq real\ (nat\ \lfloor x\rfloor)$ $x \leq real\ (nat\ \lfloor x\rfloor + 1)$ **by** *linarith+*
moreover from x **have** $nat\ \lfloor x\rfloor < N$ **by** *linarith*
ultimately have $\exists n\in\{0..<N\}.\ x \in \{real\ n..real\ (n + 1)\}$
by (*intro bexI*[*of - nat* $\lfloor x\rfloor$]) *simp-all*
thus *?thesis* **by** *blast*
qed
qed *auto*

2.2 Asymptotics of *ln-Gamma*

This is the error term that occurs in the expansion of *ln-Gamma*. It can be shown to be of order $O(s^{-n})$.

definition *stirling-integral* :: $nat \Rightarrow 'a$ :: $\{real\text{-}normed\text{-}div\text{-}algebra,\ banach\} \Rightarrow 'a$
where
stirling-integral n s =
$lim\ (\lambda N.\ integral\ \{0..N\}\ (\lambda x.\ of\text{-}real\ (pbernpoly\ n\ x)\ /\ (of\text{-}real\ x + s)\ \hat{}\ n))$

context
fixes s :: *complex* **assumes** s: $Re\ s > 0$
fixes *approx* :: $nat \Rightarrow complex$
defines $approx \equiv (\lambda N.$

($\sum n = 1..<N.\ s\ /\ \textit{of-nat}\ n - \ln\ (1 + s\ /\ \textit{of-nat}\ n)) - (\textit{euler-mascheroni} * s + \ln\ s) - (* \longrightarrow \textit{ln-Gamma}\ s\ *)$
$(\textit{ln-Gamma}\ (\textit{of-nat}\ N) - \ln\ (2 * pi\ /\ \textit{of-nat}\ N)\ /\ 2 - \textit{of-nat}\ N * \ln\ (\textit{of-nat}\ N) + \textit{of-nat}\ N) - (* \longrightarrow 0\ *)$
$s * (\textit{harm}\ (N - 1) - \ln\ (\textit{of-nat}\ (N - 1)) - \textit{euler-mascheroni}) + (* \longrightarrow 0\ *)$
$s * (\ln\ (\textit{of-nat}\ N + s) - \ln\ (\textit{of-nat}\ (N - 1))) - (* \longrightarrow 0\ *)$
$(1/2) * (\ln\ (\textit{of-nat}\ N + s) - \ln\ (\textit{of-nat}\ N)) + \quad (* \longrightarrow 0\ *)$
$\textit{of-nat}\ N * (\ln\ (\textit{of-nat}\ N + s) - \ln\ (\textit{of-nat}\ N)) - \ (* \longrightarrow s\ *)$
$(s - 1/2) * \ln\ s - \ln\ (2 * pi)\ /\ 2)$
begin

qualified lemma
assumes *N*: $N > 0$
shows *integrable-pbernpoly-1*:
$(\lambda x.\ \textit{of-real}\ (-\textit{pbernpoly}\ 1\ x)\ /\ (\textit{of-real}\ x + s))\ \textit{integrable-on}\ \{0..\textit{real}\ N\}$
and *integral-pbernpoly-1-aux*:
$\textit{integral}\ \{0..\textit{real}\ N\}\ (\lambda x.\ -\textit{of-real}\ (\textit{pbernpoly}\ 1\ x)\ /\ (\textit{of-real}\ x + s)) = \textit{approx}\ N$
and *has-integral-pbernpoly-1*:
$((\lambda x.\ \textit{pbernpoly}\ 1\ x\ /(x + s))\ \textit{has-integral}$
$(\sum m<N.\ (\textit{of-nat}\ m + 1\ /\ 2 + s) * (\ln\ (\textit{of-nat}\ m + s) -$
$\ln\ (\textit{of-nat}\ m + 1 + s)) + 1))\ \{0..\textit{real}\ N\}$
proof −
let $?A = (\lambda n.\ \{\textit{of-nat}\ n..\textit{of-nat}\ (n+1)\})$ ‘ $\{0..<N\}$
have *has-integral*:
$((\lambda x.\ -\textit{pbernpoly}\ 1\ x\ /\ (x + s))\ \textit{has-integral}$
$(\textit{of-nat}\ n + 1/2 + s) * (\ln\ (\textit{of-nat}\ (n + 1) + s) - \ln\ (\textit{of-nat}\ n + s)) - 1)$
$\{\textit{of-nat}\ n..\textit{of-nat}\ (n + 1)\}$ **for** *n*
proof (*rule has-integral-spike*)
have $((\lambda x.\ (\textit{of-nat}\ n + 1/2 + s) * (1\ /\ (x + s)) - 1)\ \textit{has-integral}$
$(\textit{of-nat}\ n + 1/2 + s) * (\ln\ (\textit{of-real}\ (\textit{real}\ (n + 1)) + s) - \ln\ (\textit{of-real}\ (\textit{real}\ n) + s)) - 1)$
$\{\textit{of-nat}\ n..\textit{of-nat}\ (n + 1)\}$
using *s has-integral-const-real*[*of 1 of-nat n of-nat* $(n + 1)$]
by (*intro has-integral-diff has-integral-mult-right fundamental-theorem-of-calculus*)
(*auto intro*!: *derivative-eq-intros has-vector-derivative-real-complex*
simp: *has-field-derivative-iff-has-vector-derivative* [*symmetric*] *field-simps*
complex-nonpos-Reals-iff)
thus $((\lambda x.\ (\textit{of-nat}\ n + 1/2 + s) * (1\ /\ (x + s)) - 1)\ \textit{has-integral}$
$(\textit{of-nat}\ n + 1/2 + s) * (\ln\ (\textit{of-nat}\ (n + 1) + s) - \ln\ (\textit{of-nat}\ n + s)) - 1)$
$\{\textit{of-nat}\ n..\textit{of-nat}\ (n + 1)\}$ **by** *simp*

show $-\textit{pbernpoly}\ 1\ x\ /\ (x + s) = (\textit{of-nat}\ n + 1/2 + s) * (1\ /\ (x + s)) - 1$
if $x \in \{\textit{of-nat}\ n..\textit{of-nat}\ (n + 1)\} - \{\textit{of-nat}\ (n + 1)\}$ **for** *x*
proof −
have *x*: $x \geq \textit{real}\ n\ x < \textit{real}\ (n + 1)$ **using** *that* **by** *simp-all*
hence $\textit{floor}\ x = \textit{int}\ n$ **by** *linarith*

moreover from *s x* **have** *complex-of-real x* $\neq -s$
by (*auto simp add*: *complex-eq-iff simp del*: *of-nat-Suc*)
ultimately show $-$*pbernpoly 1 x* / (*x* + *s*) = (*of-nat n* + *1*/*2* + *s*) $*$ (*1* / (*x* + *s*)) $-$ *1*
by (*auto simp*: *pbernpoly-def bernpoly-def frac-def divide-simps add-eq-0-iff2*)
qed
qed *simp-all*
hence $*$: $\bigwedge$*I*. *I*$\in$*?A* $\Longrightarrow$ ((λ*x*. $-$*pbernpoly 1 x* / (*x* + *s*)) *has-integral*
(*Inf I* + *1*/*2* + *s*) $*$ (*ln* (*Inf I* + *1* + *s*) $-$ *ln* (*Inf I* + *s*)) $-$ *1*) *I*
by (*auto simp*: *add-ac*)
have ((λ*x*. $-$ *pbernpoly 1 x* / (*x* + *s*)) *has-integral*
($\sum$*I*$\in$*?A*. (*Inf I* + *1* / *2* + *s*) $*$ (*ln* (*Inf I* + *1* + *s*) $-$ *ln* (*Inf I* + *s*)) $-$ *1*))
($\bigcup$*n*$\in${*0*..<*N*}. {*real n*..*real* (*n* + *1*)}) (**is** (- *has-integral ?i*) -)
apply (*intro has-integral-Union* $*$ *finite-imageI*)
apply (*force intro*!: *negligible-atLeastAtMostI*)+
done
hence *has-integral*: ((λ*x*. $-$ *pbernpoly 1 x* / (*x* + *s*)) *has-integral ?i*) {*0*..*real N*}
by (*rule has-integral-spike-set*)
(*insert Union-atLeastAtMost*[*of N*], *cases N* = *0*, *simp-all add*: *Union-atLeastAtMost*)
hence (λ*x*. $-$ *pbernpoly 1 x* / (*x* + *s*)) *integrable-on* {*0*..*real N*}
and *integral*: *integral* {*0*..*real N*} (λ*x*. $-$ *pbernpoly 1 x* / (*x* + *s*)) = *?i*
by (*simp-all add*: *has-integral-iff*)
show (λ*x*. $-$ *pbernpoly 1 x* / (*x* + *s*)) *integrable-on* {*0*..*real N*} **by** *fact*

note *has-integral-neg*[*OF has-integral*]
also have $-$*?i* = ($\sum$*x*<*N*. (*of-nat x* + *1* / *2* + *s*) $*$ (*ln* (*of-nat x* + *s*) $-$ *ln* (*of-nat x* + *1* + *s*)) + *1*)
by (*subst sum.reindex*)
(*simp-all add*: *inj-on-def atLeast0LessThan algebra-simps sum-negf* [*symmetric*])
finally show *has-integral*:
((λ*x*. *of-real* (*pbernpoly 1 x*) / (*of-real x* + *s*)) *has-integral* . . .) {*0*..*real N*} **by** *simp*

note *integral*
also have *?i* = ($\sum$*n*<*N*. (*of-nat n* + *1* / *2* + *s*) $*$
(*ln* (*of-nat n* + *1* + *s*) $-$ *ln* (*of-nat n* + *s*))) $-$ *N* (**is** - = *?S* $-$ -)
by (*subst sum.reindex*) (*simp-all add*: *inj-on-def sum-subtractf atLeast0LessThan*)
also have *?S* = ($\sum$*n*<*N*. *of-nat n* $*$ (*ln* (*of-nat n* + *1* + *s*) $-$ *ln* (*of-nat n* + *s*))) +
(*s* + *1* / *2*) $*$ ($\sum$*n*<*N*. *ln* (*of-nat* (*Suc n*) + *s*) $-$ *ln* (*of-nat n* + *s*))
(**is** - = *?S1* + - $*$ *?S2*) **by** (*simp add*: *algebra-simps sum.distrib sum-subtractf sum-distrib-left*)
also have *?S2* = *ln* (*of-nat N* + *s*) $-$ *ln s* **by** (*subst sum-lessThan-telescope*) *simp*
also have *?S1* = ($\sum$*n*=*1*..<*N*. *of-nat n* $*$ (*ln* (*of-nat n* + *1* + *s*) $-$ *ln* (*of-nat n* + *s*)))
by (*intro sum.mono-neutral-right*) *auto*

also have ... $=$ ($\sum$ *n=1..<N. of-nat n* $*$ *ln* (*of-nat n* $+$ *1* $+$ *s*)) $-$ ($\sum$ *n=1..<N. of-nat n* $*$ *ln* (*of-nat n* $+$ *s*))
  by (*simp add*: *algebra-simps sum-subtractf*)
 also have ($\sum$ *n=1..<N. of-nat n* $*$ *ln* (*of-nat n* $+$ *1* $+$ *s*)) $=$
  ($\sum$ *n=1..<N.* (*of-nat n* $-$ *1*) $*$ *ln* (*of-nat n* $+$ *s*)) $+$ (*N* $-$ *1*) $*$ *ln* (*of-nat N* $+$ *s*)
  by (*induction N*) (*simp-all add*: *add-ac of-nat-diff*)
 also have ... $-$ ($\sum$ *n* $=$ *1..<N. of-nat n* $*$ *ln* (*of-nat n* $+$ *s*)) $=$
  $-$($\sum$ *n=1..<N. ln* (*of-nat n* $+$ *s*)) $+$ (*N* $-$ *1*) $*$ *ln* (*of-nat N* $+$ *s*)
  by (*induction N*) (*simp-all add*: *algebra-simps*)
 also from *s* **have** *neq*: *s* $+$ *of-nat x* $\neq$ *0* **for** *x* **by** (*auto simp*: *complex-eq-iff*)
 hence ($\sum$ *n=1..<N. ln* (*of-nat n* $+$ *s*)) $=$ ($\sum$ *n=1..<N. ln* (*of-nat n*) $+$ *ln* (*1* $+$ *s/n*))
  by (*intro sum.cong refl*, *subst Ln-times-of-nat* [*symmetric*]) (*auto simp*: *divide-simps add-ac*)
 also have ... $=$ *ln* (*fact* (*N* $-$ *1*)) $+$ ($\sum$ *n=1..<N. ln* (*1* $+$ *s/n*))
  by (*induction N*) (*simp-all add*: *Ln-times-of-nat fact-reduce add-ac*)
 also have ($\sum$ *n=1..<N. ln* (*1* $+$ *s/n*)) $=$ $-$($\sum$ *n=1..<N. s* / *n* $-$ *ln* (*1* $+$ *s/n*)) $+$ *s* $*$ ($\sum$ *n=1..<N. 1* / *of-nat n*)
  by (*simp add*: *sum-distrib-left sum-subtractf*)
 also from *N* **have** *ln* (*fact* (*N* $-$ *1*)) $=$ *ln-Gamma* (*of-nat N* :: *complex*)
  by (*simp add*: *ln-Gamma-complex-conv-fact*)
 also have {*1..<N*} $=$ {*1..N* $-$ *1*} **by** *auto*
 hence ($\sum$ *n* $=$ *1..<N. 1* / *of-nat n*) $=$ (*harm* (*N* $-$ *1*) :: *complex*)
  by (*simp add*: *harm-def divide-simps*)
 also have $-$ (*ln-Gamma* (*of-nat N*) $+$ ($-$ ($\sum$ *n* $=$ *1..<N. s* / *of-nat n* $-$ *ln* (*1* $+$ *s* / *of-nat n*)) $+$
  s $*$ *harm* (*N* $-$ *1*))) $+$ *of-nat* (*N* $-$ *1*) $*$ *ln* (*of-nat N* $+$ *s*) $+$
  (*s* $+$ *1* / *2*) $*$ (*ln* (*of-nat N* $+$ *s*) $-$ *ln s*) $-$ *of-nat N* $=$ *approx N*
  using *N* **by** (*simp add*: *field-simps of-nat-diff ln-div approx-def Ln-of-nat*
   ln-Gamma-complex-of-real [*symmetric*])
 finally show *integral* {*0..of-nat N*} (λ*x.* $-$*of-real* (*pbernpoly 1 x*) / (*of-real x* $+$ *s*)) $=$ ...
  by *simp*
qed

lemma *integrable-ln-Gamma-aux*:
 shows (λ*x. of-real* (*pbernpoly n x*) / (*of-real x* $+$ *s*) ˆ *n*) *integrable-on* {*0..real N*}
proof (*cases n* $=$ *1*)
 case *True*
 with *s* **show** *?thesis* **using** *integrable-neg*[*OF integrable-pbernpoly-1*[*of N*]]
  by (*cases N* $=$ *0*) (*simp-all add*: *integrable-on-negligible*)
next
 case *False*
 from *s* **have** *of-real x* $+$ *s* $\neq$ *0* **if** *x* $\geq$ *0* **for** *x* **using** *that*
  by (*auto simp*: *complex-eq-iff add-eq-0-iff2*)
 with *False s* **show** *?thesis*
  by (*auto intro*!: *integrable-continuous-real continuous-intros*)

qed

This following proof is based on "Rudiments of the theory of the gamma function" by Bruce Berndt [1].

qualified lemma *integral-pbernpoly-1*:
 (λN. *integral* {*0..real N*} (λx. *pbernpoly 1 x* / (x + s)))
  $\longrightarrow$ −*ln-Gamma s* − *s* + (*s* − *1* / *2*) ∗ *ln s* + *ln* (*2* ∗ *pi*) / *2*
proof −
 have *neq*: *s* + *of-real x* ≠ *0* **if** *x* ≥ *0* **for** *x* :: *real*
  using *that s* **by** (*auto simp*: *complex-eq-iff*)
 have (*approx* $\longrightarrow$ *ln-Gamma s* − *0* − *0* + *0* − *0* + *s* − (*s* − *1/2*) ∗ *ln s* − *ln* (*2* ∗ *pi*) / *2*) *at-top*
  unfolding *approx-def*
 proof (*intro tendsto-add tendsto-diff*)
  from *s* **have** *s'*: *s* ∉ $\mathbb{Z}_{\leq 0}$ **by** (*auto elim*!: *nonpos-Ints-cases*)
  have (λn. $\sum$ *i=1..<n*. *s* / *of-nat i* − *ln* (*1* + *s* / *of-nat i*)) $\longrightarrow$
     ln-Gamma s + *euler-mascheroni* ∗ *s* + *ln s* (**is** *?f* $\longrightarrow$ -)
   using *ln-Gamma-series'-aux*[*OF s'*] **unfolding** *sums-def*
   by (*subst LIMSEQ-Suc-iff* [*symmetric*], *subst* (*asm*) *sum-atLeast1-atMost-eq* [*symmetric*])
    (*simp add*: *atLeastLessThanSuc-atLeastAtMost*)
  thus ((λn. *?f n* − (*euler-mascheroni* ∗ *s* + *ln s*)) $\longrightarrow$ *ln-Gamma s*) *at-top*
   by (*auto intro*: *tendsto-eq-intros*)
 next
  show (λx. *complex-of-real* (*ln-Gamma* (*real x*) − *ln* (*2* ∗ *pi* / *real x*) / *2* −
      real x ∗ *ln* (*real x*) + *real x*)) $\longrightarrow$ *0*
  proof (*intro tendsto-of-real-0-I*
     filterlim-compose[*OF tendsto-sandwich filterlim-real-sequentially*])
   show *eventually* (λx::*real*. *ln-Gamma x* − *ln* (*2* ∗ *pi* / *x*) / *2* − *x* ∗ *ln x* + *x* ≥ *0*) *at-top*
    using *eventually-ge-at-top*[*of 1*::*real*]
    by *eventually-elim* (*insert ln-Gamma-bounds*(*1*), *simp add*: *algebra-simps*)
   show *eventually* (λx::*real*. *ln-Gamma x* − *ln* (*2* ∗ *pi* / *x*) / *2* − *x* ∗ *ln x* + *x* ≤
     1 / *12* ∗ *inverse x*) *at-top*
    using *eventually-ge-at-top*[*of 1*::*real*]
    by *eventually-elim* (*insert ln-Gamma-bounds*(*2*), *simp add*: *field-simps*)
   show ((λx::*real*. *1* / *12* ∗ *inverse x*) $\longrightarrow$ *0*) *at-top*
    by (*intro tendsto-mult-right-zero tendsto-inverse-0-at-top filterlim-ident*)
  qed *simp-all*
 next
  have (λx. *s* ∗ *of-real* (*harm* (*x* − *1*) − *ln* (*real* (*x* − *1*)) − *euler-mascheroni*)) $\longrightarrow$
     s ∗ *of-real* (*euler-mascheroni* − *euler-mascheroni*)
   by (*subst LIMSEQ-Suc-iff* [*symmetric*], *intro tendsto-intros*)
    (*insert euler-mascheroni-LIMSEQ*, *simp-all*)
  also have *?this* ⟷ (λx. *s* ∗ (*harm* (*x* − *1*) − *ln* (*of-nat* (*x* − *1*)) − *euler-mascheroni*)) $\longrightarrow$ *0*
  by (*intro filterlim-cong refl eventually-mono*[*OF eventually-gt-at-top*[*of 1*::*nat*]])

(*auto simp*: *Ln-of-nat of-real-harm*)
finally show (λx. *s* $*$ (*harm* ($x - 1$) $-$ *ln* (*of-nat* ($x - 1$)) $-$ *euler-mascheroni*)) $\longrightarrow$ *0* **.**
next
have ((λx. *ln* (*1* + (*s* + *1*) / *of-real x*)) $\longrightarrow$ *ln* (*1* + *0*)) *at-top* (**is** *?P*)
by (*intro tendsto-intros tendsto-divide-0*[*OF tendsto-const*])
(*simp-all add*: *filterlim-ident filterlim-at-infinity-conv-norm-at-top filterlim-abs-real*)
also have *ln* (*of-real* (*x* + *1*) + *s*) $-$ *ln* (*complex-of-real x*) = *ln* (*1* + (*s* + *1*) / *of-real x*)
if *x* > *1* **for** *x* **using** *that s*
using *Ln-divide-of-real*[*of x of-real* (*x* + *1*) + *s*, *symmetric*] *neq*[*of x*+*1*]
by (*simp add*: *field-simps Ln-of-real*)
hence *?P* $\longleftrightarrow$ ((λx. *ln* (*of-real* (*x* + *1*) + *s*) $-$ *ln* (*of-real x*)) $\longrightarrow$ *0*) *at-top*
by (*intro filterlim-cong refl*)
(*auto intro*: *eventually-mono*[*OF eventually-gt-at-top*[*of 1*::*real*]])
finally have ((λn. *ln* (*of-real* (*real n* + *1*) + *s*) $-$ *ln* (*of-real* (*real n*))) $\longrightarrow$ *0*) *at-top*
by (*rule filterlim-compose*[*OF - filterlim-real-sequentially*])
hence ((λn. *ln* (*of-nat n* + *s*) $-$ *ln* (*of-nat* (*n* $-$ *1*))) $\longrightarrow$ *0*) *at-top*
by (*subst LIMSEQ-Suc-iff* [*symmetric*]) (*simp add*: *add-ac*)
thus (λx. *s* $*$ (*ln* (*of-nat x* + *s*) $-$ *ln* (*of-nat* (*x* $-$ *1*)))) $\longrightarrow$ *0*
by (*rule tendsto-mult-right-zero*)
next
have ((λx. *ln* (*1* + *s* / *of-real x*)) $\longrightarrow$ *ln* (*1* + *0*)) *at-top* (**is** *?P*)
by (*intro tendsto-intros tendsto-divide-0*[*OF tendsto-const*])
(*simp-all add*: *filterlim-ident filterlim-at-infinity-conv-norm-at-top filterlim-abs-real*)
also have *ln* (*of-real x* + *s*) $-$ *ln* (*of-real x*) = *ln* (*1* + *s* / *of-real x*) **if** *x* > *0* **for** *x*
using *Ln-divide-of-real*[*of x of-real x* + *s*] *neq*[*of x*] *that*
by (*auto simp*: *field-simps Ln-of-real*)
hence *?P* $\longleftrightarrow$ ((λx. *ln* (*of-real x* + *s*) $-$ *ln* (*of-real x*)) $\longrightarrow$ *0*) *at-top*
using *s* **by** (*intro filterlim-cong refl*)
(*auto intro*: *eventually-mono* [*OF eventually-gt-at-top*[*of 1*::*real*]])
finally have (λx. (*1*/*2*) $*$ (*ln* (*of-real* (*real x*) + *s*) $-$ *ln* (*of-real* (*real x*)))) $\longrightarrow$ *0*
by (*rule tendsto-mult-right-zero*[*OF filterlim-compose*[*OF - filterlim-real-sequentially*]])
thus (λx. (*1*/*2*) $*$ (*ln* (*of-nat x* + *s*) $-$ *ln* (*of-nat x*))) $\longrightarrow$ *0* **by** *simp*
next
have ((λx. *x* $*$ (*ln* (*1* + *s* / *of-real x*))) $\longrightarrow$ *s*) *at-top* (**is** *?P*)
by (*rule stirling-limit-aux2*)
also have *ln* (*1* + *s* / *of-real x*) = *ln* (*of-real x* + *s*) $-$ *ln* (*of-real x*) **if** *x* > *1* **for** *x*
using *that s Ln-divide-of-real* [*of x of-real x* + *s*, *symmetric*] *neq*[*of x*]
by (*auto simp*: *Ln-of-real field-simps*)
hence *?P* $\longleftrightarrow$ ((λx. *of-real x* $*$ (*ln* (*of-real x* + *s*) $-$ *ln* (*of-real x*))) $\longrightarrow$ *s*) *at-top*
by (*intro filterlim-cong refl*)
(*auto intro*: *eventually-mono*[*OF eventually-gt-at-top*[*of 1*::*real*]])

finally have (λn. *of-real* (*real n*) $*$ (*ln* (*of-real* (*real n*) + *s*) $-$ *ln* (*of-real* (*real n*)))) $\longrightarrow$ *s*
by (*rule filterlim-compose*[*OF* - *filterlim-real-sequentially*])
thus (λn. *of-nat n* $*$ (*ln* (*of-nat n* + *s*) $-$ *ln* (*of-nat n*))) $\longrightarrow$ *s* **by** *simp*
qed *simp-all*
also have *?this* $\longleftrightarrow$ ((λN. *integral* {*0..real N*} (λx. $-$*pbernpoly 1 x* / (*x* + *s*))) $\longrightarrow$
ln-Gamma s + *s* $-$ (*s* $-$ *1/2*) $*$ *ln s* $-$ *ln* (*2* $*$ *pi*) / *2*) *at-top*
using *integral-pbernpoly-1-aux*
by (*intro filterlim-cong refl*)
(*auto intro*: *eventually-mono*[*OF eventually-gt-at-top*[*of 0*::*nat*]])
also have (λN. *integral* {*0..real N*} (λx. $-$*pbernpoly 1 x* / (*x* + *s*))) =
(λN. $-$*integral* {*0..real N*} (λx. *pbernpoly 1 x* / (*x* + *s*)))
by (*simp add*: *fun-eq-iff*)
finally show *?thesis* **by** (*simp add*: *tendsto-minus-cancel-left* [*symmetric*] *algebra-simps*)
qed

qualified lemma *pbernpoly-integral-conv-pbernpoly-integral-Suc*:
assumes $n \geq 1$
shows *integral* {*0..real N*} (λx. *pbernpoly n x* / (*x* + *s*) ˆ *n*) =
of-real (*pbernpoly* (*Suc n*) (*real N*)) / (*of-nat* (*Suc n*) $*$ (*s* + *of-nat N*) ˆ *n*) $-$
of-real (*bernoulli* (*Suc n*)) / (*of-nat* (*Suc n*) $*$ *s* ˆ *n*) + *of-nat n* / *of-nat* (*Suc n*) $*$
integral {*0..real N*} (λx. *of-real* (*pbernpoly* (*Suc n*) *x*) / (*of-real x* + *s*) ˆ *Suc n*)
proof $-$
note [*derivative-intros*] = *has-field-derivative-pbernpoly-Suc′*
define *I* **where** *I* = $-$*of-real* (*pbernpoly* (*Suc n*) (*of-nat N*)) / (*of-nat* (*Suc n*) $*$ (*of-nat N* + *s*) ˆ *n*) +
of-real (*bernoulli* (*Suc n*) / *real* (*Suc n*)) / *s* ˆ *n* +
integral {*0..real N*} (λx. *of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*)
have ((λx. ($-$*of-nat n* $*$ *inverse* (*of-real x* + *s*) ˆ *Suc n*) $*$
(*of-real* (*pbernpoly* (*Suc n*) *x*) / (*of-nat* (*Suc n*))))
has-integral $-$*I*) {*0..real N*}
proof (*rule integration-by-parts-interior-strong*[*OF bounded-bilinear-mult*])
fix *x* :: *real* **assume** *x*: *x* $\in$ {*0*<..<*real N*} $-$ *real* ‘ {*0..N*}
have $x \notin \mathbb{Z}$
proof
assume $x \in \mathbb{Z}$
then obtain *n* **where** *x* = *of-int n* **by** (*auto elim*!: *Ints-cases*)
with *x* **have** *x′*: *x* = *of-nat* (*nat n*) **by** *simp*
from *x* **show** *False* **by** (*auto simp*: *x′*)
qed
hence ((λx. *of-real* (*pbernpoly* (*Suc n*) *x* / *of-nat* (*Suc n*))) *has-vector-derivative*
complex-of-real (*pbernpoly n x*)) (*at x*)
by (*intro has-vector-derivative-of-real*) (*auto intro*!: *derivative-eq-intros*)
thus ((λx. *of-real* (*pbernpoly* (*Suc n*) *x*) / *of-nat* (*Suc n*)) *has-vector-derivative*

complex-of-real (*pbernpoly n x*)) (*at x*) **by** *simp*
from *x s* **have** *complex-of-real x* + *s* ≠ *0* **by** (*auto simp*: *complex-eq-iff*)
thus ((λ*x*. *inverse* (*of-real x* + *s*) ˆ *n*) *has-vector-derivative*
− *of-nat n* ∗ *inverse* (*of-real x* + *s*) ˆ *Suc n*) (*at x*) **using** *x s assms*
by (*auto intro*!: *derivative-eq-intros has-vector-derivative-real-complex simp*: *divide-simps power-add* [*symmetric*]
simp del: *power-Suc*)
next
have *complex-of-real x* + *s* ≠ *0* **if** *x* ≥ *0* **for** *x*
using *that s* **by** (*auto simp*: *complex-eq-iff*)
thus *continuous-on* {*0*..*real N*} (λ*x*. *inverse* (*of-real x* + *s*) ˆ *n*)
continuous-on {*0*..*real N*} (λ*x*. *complex-of-real* (*pbernpoly* (*Suc n*) *x*) / *of-nat* (*Suc n*))
using *assms s* **by** (*auto intro*!: *continuous-intros simp del*: *of-nat-Suc*)
next
have ((λ*x*. *inverse* (*of-real x* + *s*) ˆ *n* ∗ *of-real* (*pbernpoly n x*)) *has-integral*
pbernpoly (*Suc n*) (*of-nat N*) / (*of-nat* (*Suc n*) ∗ (*of-nat N* + *s*) ˆ *n*) −
of-real (*bernoulli* (*Suc n*) / *real* (*Suc n*)) / *s* ˆ *n* − −*I*) {*0*..*real N*}
using *integrable-ln-Gamma-aux*[*of n N*] *assms*
by (*auto simp*: *I-def has-integral-integral divide-simps*)
thus ((λ*x*. *inverse* (*of-real x* + *s*) ˆ *n* ∗ *of-real* (*pbernpoly n x*)) *has-integral*
inverse (*of-real* (*real N*) + *s*) ˆ *n* ∗ (*of-real* (*pbernpoly* (*Suc n*) (*real N*)) /
of-nat (*Suc n*)) −
inverse (*of-real 0* + *s*) ˆ *n* ∗ (*of-real* (*pbernpoly* (*Suc n*) *0*) / *of-nat* (*Suc n*)) − − *I*)
{*0*..*real N*} **by** (*simp-all add*: *field-simps*)
qed *simp-all*
also have (λ*x*. − *of-nat n* ∗ *inverse* (*of-real x* + *s*) ˆ *Suc n* ∗ (*of-real* (*pbernpoly* (*Suc n*) *x*) /
of-nat (*Suc n*))) =
(λ*x*. − (*of-nat n* / *of-nat* (*Suc n*) ∗ *of-real* (*pbernpoly* (*Suc n*) *x*) /
(*of-real x* + *s*) ˆ *Suc n*))
by (*simp add*: *divide-simps fun-eq-iff*)
finally have ((λ*x*. − (*of-nat n* / *of-nat* (*Suc n*) ∗ *of-real* (*pbernpoly* (*Suc n*) *x*) /
(*of-real x* + *s*) ˆ *Suc n*)) *has-integral* − *I*) {*0*..*real N*} .
from *has-integral-neg*[*OF this*] **show** *?thesis*
by (*auto simp add*: *I-def has-integral-iff algebra-simps integral-mult-right* [*symmetric*]
simp del: *power-Suc of-nat-Suc*)
qed

lemma *pbernpoly-over-power-tendsto-0*:
assumes *n* > *0*
shows (λ*x*. *of-real* (*pbernpoly* (*Suc n*) (*real x*)) / (*of-nat* (*Suc n*) ∗ (*s* + *of-nat x*) ˆ *n*)) ⟶ *0*
proof −
from *s* **have** *neq*: *s* + *of-nat n* ≠ *0* **for** *n* **by** (*auto simp*: *complex-eq-iff*)

from *bounded-pbernpoly*[*of Suc n*] **guess** *c* **. note** *c* = *this*
have *eventually* (*λx. norm* (*of-real* (*pbernpoly* (*Suc n*) (*real x*)) /
(*of-nat* (*Suc n*) ∗ (*s* + *of-nat x*) ˆ *n*)) ≤
(*c* / *real* (*Suc n*)) / *real x* ˆ *n*) *at-top*
using *eventually-gt-at-top*[*of 0*::*nat*]
proof *eventually-elim*
case (*elim x*)
have *norm* (*of-real* (*pbernpoly* (*Suc n*) (*real x*)) /
(*of-nat* (*Suc n*) ∗ (*s* + *of-nat x*) ˆ *n*)) ≤
(*c* / *real* (*Suc n*)) / *norm* (*s* + *of-nat x*) ˆ *n* (**is** - ≤ *?rhs*) **using** *c*[*of x*]
by (*auto simp*: *norm-divide norm-mult norm-power neq field-simps simp del*: *of-nat-Suc*)
also have *real x* ≤ *cmod* (*s* + *of-nat x*)
using *complex-Re-le-cmod*[*of s* + *of-nat x*] *s* **by** *simp*
hence *?rhs* ≤ (*c* / *real* (*Suc n*)) / *real x* ˆ *n* **using** *s elim c*[*of 0*] *neq*[*of x*]
by (*intro divide-left-mono power-mono mult-pos-pos divide-nonneg-pos zero-less-power*) *auto*
finally show *?case* **.**
qed
moreover have (*λx.* (*c* / *real* (*Suc n*)) / *real x* ˆ *n*) ⟶ *0*
by (*intro real-tendsto-divide-at-top*[*OF tendsto-const*] *filterlim-pow-at-top assms filterlim-real-sequentially*)
ultimately show *?thesis* **by** (*rule Lim-null-comparison*)
qed

lemma *convergent-stirling-integral*:
assumes *n* > *0*
shows *convergent* (*λN. integral* {*0..real N*}
(*λx. of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*)) (**is** *convergent* (*?f n*))
proof −
have *convergent* (*?f* (*Suc n*)) **for** *n*
proof (*induction n*)
case *0*
thus *?case* **using** *integral-pbernpoly-1* **by** (*auto intro*!: *convergentI*)
next
case (*Suc n*)
have *convergent* (*λN. ?f* (*Suc n*) *N* −
of-real (*pbernpoly* (*Suc* (*Suc n*)) (*real N*)) /
(*of-nat* (*Suc* (*Suc n*)) ∗ (*s* + *of-nat N*) ˆ *Suc n*) +
of-real (*bernoulli* (*Suc* (*Suc n*)) / (*real* (*Suc* (*Suc n*)))) / *s* ˆ *Suc n*)
(**is** *convergent ?g*)
by (*intro convergent-add convergent-diff Suc convergent-const convergentI*[*OF pbernpoly-over-power-tendsto-0*]) *simp-all*
also have *?g* = (*λN. of-nat* (*Suc n*) / *of-nat* (*Suc* (*Suc n*)) ∗ *?f* (*Suc* (*Suc n*)) *N*) **using** *s*
by (*subst pbernpoly-integral-conv-pbernpoly-integral-Suc*)
(*auto simp*: *fun-eq-iff field-simps simp del*: *of-nat-Suc power-Suc*)
also have *convergent* . . . ⟷ *convergent* (*?f* (*Suc* (*Suc n*)))
by (*intro convergent-mult-const-iff*) (*simp-all del*: *of-nat-Suc*)

finally show *?case* **.**
qed
from *this*[*of n − 1*] *assms* **show** *?thesis* **by** *simp*
qed

lemma *stirling-integral-conv-stirling-integral-Suc*:
assumes $n > 0$
shows *stirling-integral n s* =
of-nat n / *of-nat* (*Suc n*) ∗ *stirling-integral* (*Suc n*) *s* −
of-real (*bernoulli* (*Suc n*)) / (*of-nat* (*Suc n*) ∗ *s* ˆ *n*)
proof −
have (λ*N*. *of-real* (*pbernpoly* (*Suc n*) (*real N*)) / (*of-nat* (*Suc n*) ∗ (*s* + *of-nat* *N*) ˆ *n*) −
of-real (*bernoulli* (*Suc n*)) / (*real* (*Suc n*) ∗ *s* ˆ *n*) +
integral {*0..real N*} (λ*x*. *of-nat n* / *of-nat* (*Suc n*) ∗
(*of-real* (*pbernpoly* (*Suc n*) *x*) / (*of-real x* + *s*) ˆ *Suc n*)))
⟶ *0* − *of-real* (*bernoulli* (*Suc n*)) / (*of-nat* (*Suc n*) ∗ *s* ˆ *n*) +
of-nat n / *of-nat* (*Suc n*) ∗ *stirling-integral* (*Suc n*) *s* (**is** *?f* ⟶ -)
unfolding *stirling-integral-def integral-mult-right*
using *convergent-stirling-integral*[*of Suc n*] *assms s*
by (*intro tendsto-intros pbernpoly-over-power-tendsto-0*)
(*auto simp*: *convergent-LIMSEQ-iff simp del*: *of-nat-Suc*)
also have *?this* ⟷ (λ*N*. *integral* {*0..real N*}
(λ*x*. *of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*)) ⟶
of-nat n / *of-nat* (*Suc n*) ∗ *stirling-integral* (*Suc n*) *s* −
of-real (*bernoulli* (*Suc n*)) / (*of-nat* (*Suc n*) ∗ *s* ˆ *n*)
using *eventually-gt-at-top*[*of 0*::*nat*] *pbernpoly-integral-conv-pbernpoly-integral-Suc*[*of n*]
assms **unfolding** *integral-mult-right*
by (*intro filterlim-cong refl*) (*auto elim*!: *eventually-mono simp del*: *power-Suc*)
finally show *?thesis* **unfolding** *stirling-integral-def*[*of n*] **by** (*rule limI*)
qed

lemma *stirling-integral-1-unfold*:
assumes $m > 0$
shows *stirling-integral 1 s* = *stirling-integral m s* / *of-nat m* −
(∑*k*=*1*..<*m*. *of-real* (*bernoulli* (*Suc k*)) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗ *s* ˆ *k*))
proof −
have *stirling-integral 1 s* = *stirling-integral* (*Suc m*) *s* / *of-nat* (*Suc m*) −
(∑*k*=*1*..<*Suc m*. *of-real* (*bernoulli* (*Suc k*)) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗ *s* ˆ *k*)) **for** *m*
proof (*induction m*)
case (*Suc m*)
let *?C* = (∑ *k* = *1*..<*Suc m*. *of-real* (*bernoulli* (*Suc k*)) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗ *s* ˆ *k*))
note *Suc.IH*
also have *stirling-integral* (*Suc m*) *s* / *of-nat* (*Suc m*) =

stirling-integral (*Suc* (*Suc m*)) *s* / *of-nat* (*Suc* (*Suc m*)) −
of-real (*bernoulli* (*Suc* (*Suc m*))) /
(*of-nat* (*Suc m*) ∗ *of-nat* (*Suc* (*Suc m*)) ∗ *s* ˆ *Suc m*)
(**is** - = *?A* − *?B*) **by** (*subst stirling-integral-conv-stirling-integral-Suc*)
(*simp-all del*: *of-nat-Suc power-Suc add*: *divide-simps*)
also have *?A* − *?B* − *?C* = *?A* − (*?B* + *?C*) **by** (*rule diff-diff-eq*)
also have *?B* + *?C* = ($\sum$ *k* = *1*..<*Suc* (*Suc m*). *of-real* (*bernoulli* (*Suc k*)) /
(*of-nat k* ∗ *of-nat* (*Suc k*) ∗ *s* ˆ *k*))
using *s* **by** (*simp add*: *divide-simps*)
finally show *?case* **.**
qed *simp-all*
note *this*[*of m* − *1*]
also from *assms* **have** *Suc* (*m* − *1*) = *m* **by** *simp*
finally show *?thesis* **.**
qed

lemma *ln-Gamma-stirling-complex*:
assumes *m* > *0*
shows *ln-Gamma s* = (*s* − *1* / *2*) ∗ *ln s* − *s* + *ln* (*2* ∗ *pi*) / *2* +
($\sum$ *k*=*1*..<*m*. *of-real* (*bernoulli* (*Suc k*)) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗
s ˆ *k*)) −
stirling-integral m s / *of-nat m*
proof −
have *ln-Gamma s* = (*s* − *1* / *2*) ∗ *ln s* − *s* + *ln* (*2* ∗ *pi*) / *2* − *stirling-integral*
1 s
using *limI*[*OF integral-pbernpoly-1*] **by** (*simp add*: *stirling-integral-def algebra-simps*)
also have *stirling-integral 1 s* = *stirling-integral m s* / *of-nat m* −
($\sum$ *k* = *1*..<*m*. *of-real* (*bernoulli* (*Suc k*)) / (*of-nat k* ∗ *of-nat* (*Suc*
k) ∗ *s* ˆ *k*))
using *assms* **by** (*rule stirling-integral-1-unfold*)
finally show *?thesis* **by** *simp*
qed

lemma *LIMSEQ-stirling-integral*:
n > *0* ⟹ (λ*x*. *integral* {*0*..*real x*} (λ*x*. *of-real* (*pbernpoly n x*) / (*of-real x* + *s*)
ˆ *n*))
⟶ *stirling-integral n s* **unfolding** *stirling-integral-def*
using *convergent-stirling-integral*[*of n*] **by** (*simp only*: *convergent-LIMSEQ-iff*)

end

lemmas *has-integral-of-real* = *has-integral-linear*[*OF* - *bounded-linear-of-real*, *unfolded o-def*]
lemmas *integral-of-real* = *integral-linear*[*OF* - *bounded-linear-of-real*, *unfolded o-def*]

lemma *integrable-ln-Gamma-aux-real*:
assumes *0* < *s*
shows (λ*x*. *pbernpoly n x* / (*x* + *s*) ˆ *n*) *integrable-on* {*0*..*real N*}
proof −

have (λx. *complex-of-real* (*pbernpoly n x* / (x + s) ˆ n)) *integrable-on* {0..*real N*}
  using *integrable-ln-Gamma-aux*[*of of-real s n N*] *assms* **by** *simp*
from *integrable-linear*[*OF this bounded-linear-Re*] **show** *?thesis*
  by (*simp only*: *o-def Re-complex-of-real*)
qed

lemma
assumes x > 0 n > 0
shows *stirling-integral-complex-of-real*:
    stirling-integral n (*complex-of-real x*) = *of-real* (*stirling-integral n x*)
  and *LIMSEQ-stirling-integral-real*:
    (λN. *integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n))
    ⟶ *stirling-integral n x*
  and *stirling-integral-real-convergent*:
    convergent (λN. *integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n))
proof −
have (λN. *integral* {0..*real N*} (λt. *of-real* (*pbernpoly n t* / (t + x) ˆ n)))
    ⟶ *stirling-integral n* (*complex-of-real x*)
  using *LIMSEQ-stirling-integral*[*of complex-of-real x n*] *assms* **by** *simp*
hence (λN. *of-real* (*integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n)))
    ⟶ *stirling-integral n* (*complex-of-real x*)
  using *integrable-ln-Gamma-aux-real*[*OF assms*(1), *of n*]
  by (*subst* (*asm*) *integral-of-real*) *simp*
from *tendsto-Re*[*OF this*]
  have (λN. *integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n))
    ⟶ *Re* (*stirling-integral n* (*complex-of-real x*)) **by** *simp*
thus *convergent* (λN. *integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n))
  by (*rule convergentI*)
thus (λN. *integral* {0..*real N*} (λt. *pbernpoly n t* / (t + x) ˆ n))
    ⟶ *stirling-integral n x* **unfolding** *stirling-integral-def*
  by (*simp add*: *convergent-LIMSEQ-iff*)
from *tendsto-of-real*[*OF this*, **where** ′a = *complex*]
    integrable-ln-Gamma-aux-real[*OF assms*(1), *of n*]
  have (λxa. *integral* {0..*real xa*}
        (λxa. *complex-of-real* (*pbernpoly n xa*) / (*complex-of-real xa* + x) ˆ n))
    ⟶ *complex-of-real* (*stirling-integral n x*)
  by (*subst* (*asm*) *integral-of-real* [*symmetric*]) *simp-all*
from *LIMSEQ-unique*[*OF this LIMSEQ-stirling-integral*[*of complex-of-real x n*]] *assms*
  show *stirling-integral n* (*complex-of-real x*) = *of-real* (*stirling-integral n x*) **by** *simp*
qed

lemma *ln-Gamma-stirling-real*:
assumes x > (0 :: *real*) m > (0::*nat*)
shows *ln-Gamma x* = (x − 1 / 2) ∗ *ln x* − x + *ln* (2 ∗ *pi*) / 2 +
    (∑ k = 1..<m. *bernoulli* (*Suc k*) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗ x ˆ k))

−
stirling-integral m x / of-nat m

proof −

from *assms* **have** *complex-of-real* (*ln-Gamma x*) = *ln-Gamma* (*complex-of-real x*)

by (*simp add*: *ln-Gamma-complex-of-real*)

also have *ln-Gamma* (*complex-of-real x*) = *complex-of-real* (
(*x* − *1* / *2*) ∗ *ln x* − *x* + *ln* (*2* ∗ *pi*) / *2* +
($\sum$ *k* = *1*..<*m*. *bernoulli* (*Suc k*) / (*of-nat k* ∗ *of-nat* (*Suc k*) ∗ *x* ˆ *k*)) −
stirling-integral m x / *of-nat m*) **using** *assms*

by (*subst ln-Gamma-stirling-complex*[*of - m*])
(*simp-all add*: *Ln-of-real stirling-integral-complex-of-real*)

finally show *?thesis* **by** (*subst* (*asm*) *of-real-eq-iff*)

qed

context
begin

private lemma *stirling-integral-bound-aux*:

assumes *n*: *n* > (*1*::*nat*)

obtains *c* **where** $\bigwedge$*s*. *Re s* > *0* ⟹ *norm* (*stirling-integral n s*) ≤ *c* / *Re s* ˆ (*n* − *1*)

proof −

obtain *c* **where** *c*: *norm* (*pbernpoly n x*) ≤ *c* **for** *x* **by** (*rule bounded-pbernpoly*[*of n*]) *blast*

have *c'*: *pbernpoly n x* ≤ *c* **for** *x* **using** *c*[*of x*] **by** (*simp add*: *abs-real-def split*: *if-splits*)

from *c*[*of 0*] **have** *c-nonneg*: *c* ≥ *0* **by** *simp*

have *norm* (*stirling-integral n s*) ≤ *c* / (*real n* − *1*) / *Re s* ˆ (*n* − *1*) **if** *s*: *Re s* > *0* **for** *s*

proof (*rule Lim-norm-ubound*[*OF - LIMSEQ-stirling-integral*])

have *pos*: *x* + *norm s* > *0* **if** *x* ≥ *0* **for** *x* **using** *s that* **by** (*intro add-nonneg-pos*) *auto*

have *nz*: *of-real x* + *s* ≠ *0* **if** *x* ≥ *0* **for** *x* **using** *s that* **by** (*auto simp*: *complex-eq-iff*)

let *?bound* = λ*N*. *c* / (*Re s* ˆ (*n* − *1*) ∗ (*real n* − *1*)) −
c / ((*real N* + *Re s*) ˆ (*n* − *1*) ∗ (*real n* − *1*))

show *eventually* (λ*N*. *norm* (*integral* {*0*..*real N*}
(λ*x*. *of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*)) ≤
c / (*real n* − *1*) / *Re s* ˆ (*n* − *1*)) *at-top*

using *eventually-gt-at-top*[*of 0*::*nat*]

proof *eventually-elim*

case (*elim N*)

let *?F* = λ*x*. −*c* / ((*x* + *Re s*) ˆ (*n* − *1*) ∗ (*real n* − *1*))

from *n s* **have** ((λ*x*. *c* / (*x* + *Re s*) ˆ *n*) *has-integral* (*?F* (*real N*) − *?F 0*)) {*0*..*real N*}

by (*intro fundamental-theorem-of-calculus*)

(*auto intro*!: *derivative-eq-intros simp*: *divide-simps power-diff add-eq-0-iff2 has-field-derivative-iff-has-vector-derivative* [*symmetric*])
also have *?F* (*real N*) − *?F 0* = *?bound N* **by** *simp*
finally have ∗: ((λ*x*. *c* / (*x* + *Re s*) ˆ *n*) *has-integral ?bound N*) {*0*..*real N*}
.
have *norm* (*integral* {*0*..*real N*} (λ*x*. *of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*)) ≤
integral {*0*..*real N*} (λ*x*. *c* / (*x* + *Re s*) ˆ *n*)
proof (*intro integral-norm-bound-integral integrable-ln-Gamma-aux s ballI*)
fix *x* **assume** *x*: *x* ∈ {*0*..*real N*}
have *norm* (*of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*) ≤ *c* / *norm* (*of-real x* + *s*) ˆ *n*
unfolding *norm-divide norm-power* **using** *c* **by** (*intro divide-right-mono*) *simp-all*
also have . . . ≤ *c* / (*x* + *Re s*) ˆ *n*
using *x c c-nonneg s nz*[*of x*] *complex-Re-le-cmod*[*of of-real x* + *s*]
by (*intro divide-left-mono power-mono mult-pos-pos zero-less-power add-nonneg-pos*) *auto*
finally show *norm* (*of-real* (*pbernpoly n x*) / (*of-real x* + *s*) ˆ *n*) ≤ . . . .
qed (*insert n s* ∗ *pos nz c*, *auto*)
also have . . . = *?bound N* **using** ∗ **by** (*simp add*: *has-integral-iff*)
also have . . . ≤ *c* / (*Re s* ˆ (*n* − *1*) ∗ (*real n* − *1*)) **using** *c-nonneg elim s n* **by** *simp*
also have . . . = *c* / (*real n* − *1*) / (*Re s* ˆ (*n* − *1*)) **by** *simp*
finally show *norm* (*integral* {*0*..*real N*} (λ*x*. *of-real* (*pbernpoly n x*) /
(*of-real x* + *s*) ˆ *n*)) ≤ *c* / (*real n* − *1*) / *Re s* ˆ (*n* − *1*) .
qed
qed (*insert s n*, *simp-all*)
thus *?thesis* **by** (*rule that*)
qed

lemma *stirling-integral-bound*:
assumes *n* > *0*
obtains *c* **where**
⋀*s*. *Re s* > *0* ⟹ *norm* (*stirling-integral n s*) ≤ *c* / *Re s* ˆ *n*
proof −
let *?f* = λ*s*. *of-nat n* / *of-nat* (*Suc n*) ∗ *stirling-integral* (*Suc n*) *s* −
of-real (*bernoulli* (*Suc n*)) / (*of-nat* (*Suc n*) ∗ *s* ˆ *n*)
from *stirling-integral-bound-aux*[*of Suc n*] *assms* **obtain** *c* **where**
c: ⋀*s*. *Re s* > *0* ⟹ *norm* (*stirling-integral* (*Suc n*) *s*) ≤ *c* / *Re s* ˆ *n* **by** *auto*
define *c1* **where** *c1* = *real n* / *real* (*Suc n*) ∗ *c*
define *c2* **where** *c2* = |*bernoulli* (*Suc n*)| / *real* (*Suc n*)
have *c2-nonneg*: *c2* ≥ *0* **by** (*simp add*: *c2-def*)
show *?thesis*
proof (*rule that*)
fix *s* :: *complex* **assume** *s*: *Re s* > *0*
have *stirling-integral n s* = *?f s* **using** *s assms*
by (*rule stirling-integral-conv-stirling-integral-Suc*)
also have *norm* . . . ≤ *norm* (*of-nat n* / *of-nat* (*Suc n*) ∗ *stirling-integral* (*Suc*

n) s) +
norm (of-real (bernoulli (Suc n)) / (of-nat (Suc n) * s ^ n))
by (rule norm-triangle-ineq4)
also have ... = real n / real (Suc n) * norm (stirling-integral (Suc n) s) +
c2 / norm s ^ n (is - = ?A + ?B)
by (simp add: norm-divide norm-mult norm-power c2-def field-simps del: of-nat-Suc)
also have ?A ≤ real n / real (Suc n) * (c / Re s ^ n)
by (intro mult-left-mono c s) simp-all
also have ... = c1 / Re s ^ n by (simp add: c1-def)
also have c2 / norm s ^ n ≤ c2 / Re s ^ n using s c2-nonneg
by (intro divide-left-mono power-mono complex-Re-le-cmod mult-pos-pos zero-less-power) auto
also have c1 / Re s ^ n + c2 / Re s ^ n = (c1 + c2) / Re s ^ n
using s by (simp add: field-simps)
finally show norm (stirling-integral n s) ≤ (c1 + c2) / Re s ^ n by − simp-all
qed
qed

end

lemma stirling-integral-holomorphic [holomorphic-intros]:
assumes m: m > 0 and ∀ s∈A. Re s > 0
shows stirling-integral m holomorphic-on A
proof −
let ?f = λs::complex. of-nat m * ((s − 1 / 2) * Ln s − s + of-real (ln (2 * pi) / 2) +
(∑ k=1..<m. of-real (bernoulli (Suc k)) / (of-nat k * of-nat (Suc k) * s ^ k)) −
ln-Gamma s)
have ?f holomorphic-on A using assms
by (auto intro!: holomorphic-intros simp del: of-nat-Suc elim!: nonpos-Reals-cases)
also have ?this ⟷ stirling-integral m holomorphic-on A
using assms by (intro holomorphic-cong refl)
(simp-all add: field-simps ln-Gamma-stirling-complex)
finally show stirling-integral m holomorphic-on A .
qed

lemma stirling-integral-continuous-on [continuous-intros]:
assumes m: m > 0 and ∀ s∈A. Re s > 0
shows continuous-on A (stirling-integral m)
by (intro holomorphic-on-imp-continuous-on stirling-integral-holomorphic assms)

lemma has-field-derivative-stirling-integral:
assumes Re x > 0 n > 0
shows (stirling-integral n has-field-derivative deriv (stirling-integral n) x) (at x)
using assms

by (*intro holomorphic-derivI*[*OF stirling-integral-holomorphic*, *of n* {*s*. *Re s* > *0*}]
open-halfspace-Re-gt) *auto*

lemma
assumes *n*: *n* > *0* **and** *x* > *0*
shows *deriv-stirling-integral-complex-of-real*:
(*deriv* ^^ *j*) (*stirling-integral n*) (*complex-of-real x*) =
complex-of-real ((*deriv* ^^ *j*) (*stirling-integral n*) *x*) (**is** *?lhs x* = *?rhs x*)
and *differentiable-stirling-integral-real*:
(*deriv* ^^ *j*) (*stirling-integral n*) *field-differentiable at x* (**is** *?thesis2*)
proof −
let *?A* = {*s*. *Re s* > *0*}
let *?f* = λ*j x*. (*deriv* ^^ *j*) (*stirling-integral n*) (*complex-of-real x*)
let *?f'* = λ*j x*. *complex-of-real* ((*deriv* ^^ *j*) (*stirling-integral n*) *x*)

have [*simp*]: *open ?A* **by** (*simp add*: *open-halfspace-Re-gt*)

have *?lhs x* = *?rhs x* ∧ (*deriv* ^^ *j*) (*stirling-integral n*) *field-differentiable at x*
if *x* > *0* **for** *x* **using** *that*
proof (*induction j arbitrary*: *x*)
case *0*
have ((λ*x*. *Re* (*stirling-integral n* (*of-real x*))) *has-field-derivative*
Re (*deriv* (λ*x*. *stirling-integral n x*) (*of-real x*))) (*at x*) **using** *0 n*
by (*auto intro*!: *derivative-intros has-vector-derivative-real-complex*
field-differentiable-derivI holomorphic-on-imp-differentiable-at[*of* - *?A*]
stirling-integral-holomorphic)
also have *?this* ⟷ (*stirling-integral n has-field-derivative*
Re (*deriv* (λ*x*. *stirling-integral n x*) (*of-real x*))) (*at x*)
using *eventually-nhds-in-open*[*of* {*0*<..} *x*] *0 n*
by (*intro has-field-derivative-cong-ev refl*)
(*auto elim*!: *eventually-mono simp*: *stirling-integral-complex-of-real*)
finally have *stirling-integral n field-differentiable at x*
by (*auto simp*: *field-differentiable-def*)
with *0 n* **show** *?case* **by** (*auto simp*: *stirling-integral-complex-of-real*)
next
case (*Suc j x*)
note *IH* = *conjunct1*[*OF Suc.IH*] *conjunct2*[*OF Suc.IH*]
have ∗: (*deriv* ^^ *Suc j*) (*stirling-integral n*) (*complex-of-real x*) =
of-real ((*deriv* ^^ *Suc j*) (*stirling-integral n*) *x*) **if** *x*: *x* > *0* **for** *x*
proof −
have *deriv* ((*deriv* ^^ *j*) (*stirling-integral n*)) (*complex-of-real x*) =
vector-derivative (λ*x*. (*deriv* ^^ *j*) (*stirling-integral n*) (*of-real x*)) (*at x*)
using *n x*
by (*intro vector-derivative-of-real-right* [*symmetric*]
holomorphic-on-imp-differentiable-at[*of* - *?A*] *holomorphic-higher-deriv*

stirling-integral-holomorphic) *auto*
also have ... = *vector-derivative* (λx. *of-real* ((*deriv* ^^ *j*) (*stirling-integral* *n*) *x*)) (*at x*)
using *eventually-nhds-in-open*[*of* {*0*<..} *x*] *x*
by (*intro vector-derivative-cong-eq*) (*auto elim*!: *eventually-mono simp*: *IH*(*1*))
also have ... = *of-real* (*deriv* ((*deriv* ^^ *j*) (*stirling-integral n*)) *x*)
by (*intro vector-derivative-of-real-left holomorphic-on-imp-differentiable-at*[*of* - *?A*]
field-differentiable-imp-differentiable IH(*2*) *x*)
finally show *?thesis* **by** *simp*
qed
have ((λx. *Re* ((*deriv* ^^ *Suc j*) (*stirling-integral n*) (*of-real x*))) *has-field-derivative*

Re (*deriv* ((*deriv* ^^ *Suc j*) (*stirling-integral n*)) (*of-real x*))) (*at x*)
using *Suc.prems n*
by (*intro derivative-intros has-vector-derivative-real-complex field-differentiable-derivI*
holomorphic-on-imp-differentiable-at[*of* - *?A*] *stirling-integral-holomorphic*
holomorphic-higher-deriv) *auto*
also have *?this* $\longleftrightarrow$ ((*deriv* ^^ *Suc j*) (*stirling-integral n*) *has-field-derivative*
Re (*deriv* ((*deriv* ^^ *Suc j*) (*stirling-integral n*)) (*of-real x*))) (*at x*)
using *eventually-nhds-in-open*[*of* {*0*<..} *x*] *Suc.prems* *
by (*intro has-field-derivative-cong-ev refl*) (*auto elim*!: *eventually-mono*)
finally have (*deriv* ^^ *Suc j*) (*stirling-integral n*) *field-differentiable at x*
by (*auto simp*: *field-differentiable-def*)
with *[*OF Suc.prems*] **show** *?case* **by** *blast*
qed
from *this*[*OF assms*(*2*)] **show** *?lhs x* = *?rhs x ?thesis2* **by** *blast+*
qed

Unfortunately, asymptotic power series cannot, in general, be differentiated. However, since *ln-Gamma* is holomorphic on the entire positive real half-space, we can differentiate its asymptotic expansion after all.

To do this, we use an ad-hoc version of the more general approach outlined in Erdelyi's "Asymptotic Expansions" for holomorphic functions: We bound the value of the j-th derivative of the remainder term at some value x by applying Cauchy's integral formula along a circle centred at x with radius $\frac{1}{2}x$.

lemma *deriv-stirling-integral-real-bound*:
assumes *m*: *m* > *0*
shows (*deriv* ^^ *j*) (*stirling-integral m*) ∈ *O*(λx::*real*. *1* / *x* ^ (*m* + *j*))
proof −
from *stirling-integral-bound*[*OF m*] **guess** *c* **. note** *c* = *this*
have *0* ≤ *cmod* (*stirling-integral m 1*) **by** *simp*
also have ... ≤ *c* **using** *c*[*of 1*] **by** *simp*
finally have *c-nonneg*: *c* ≥ *0* **.**
define *B* **where** *B* = *c* * *2* ^ (*m* + *Suc j*)
define *B'* **where** *B'* = *B* * *fact j* / *2*

```
  have eventually (λx::real. norm ((deriv ^^ j) (stirling-integral m) x) ≤
        B' * norm (1 / x ^ (m+ j))) at-top
    using eventually-gt-at-top[of 0::real]
  proof eventually-elim
    case (elim x)
    have Re s > 0 if s ∈ cball (of-real x) (x/2) for s :: complex
    proof −
      have x − Re s ≤ norm (of-real x − s) using complex-Re-le-cmod[of of-real
x − s] by simp
      also from that have ... ≤ x/2 by (simp add: dist-complex-def)
      finally show ?thesis using elim by simp
    qed
    hence ((λu. stirling-integral m u / (u − of-real x) ^ Suc j) has-contour-integral
            complex-of-real (2 * pi) * i / fact j *
            (deriv ^^ j) (stirling-integral m) (of-real x)) (circlepath (of-real x) (x/2))
      using m elim
      by (intro Cauchy-has-contour-integral-higher-derivative-circlepath
              stirling-integral-continuous-on stirling-integral-holomorphic) auto
    hence norm (of-real (2 * pi) * i / fact j * (deriv ^^ j) (stirling-integral m)
(of-real x)) ≤
            B / x ^ (m + Suc j) * (2 * pi * (x / 2))
    proof (rule has-contour-integral-bound-circlepath)
      fix u :: complex assume dist: norm (u − of-real x) = x / 2
      have Re (of-real x − u) ≤ norm (of-real x − u) by (rule complex-Re-le-cmod)
      also have ... = x / 2 using dist by (simp add: norm-minus-commute)
      finally have Re-u: Re u ≥ x/2 using elim by simp
      have norm (stirling-integral m u / (u − of-real x) ^ Suc j) ≤
              c / Re u ^ m / (x / 2) ^ Suc j using Re-u elim
        unfolding norm-divide norm-power dist
        by (intro divide-right-mono zero-le-power c) simp-all
      also have ... ≤ c / (x/2) ^ m / (x / 2) ^ Suc j using c-nonneg elim Re-u
        by (intro divide-right-mono divide-left-mono power-mono) simp-all
      also have ... = B / x ^ (m + Suc j) using elim by (simp add: B-def
field-simps power-add)
      finally show norm (stirling-integral m u / (u − of-real x) ^ Suc j) ≤ B / x
^ (m + Suc j) .
    qed (insert elim c-nonneg, auto simp: B-def simp del: power-Suc)
    hence cmod ((deriv ^^ j) (stirling-integral m) (of-real x)) ≤ B' / x ^ (j + m)
      using elim by (simp add: field-simps norm-divide norm-mult norm-power
B'-def)
    with elim m show ?case by (simp-all add: add-ac deriv-stirling-integral-complex-of-real)
  qed
  thus ?thesis by (rule bigoI)
qed

definition stirling-sum where
  stirling-sum j m x =
    (−1) ^ j * (∑ k = 1..<m. (of-real (bernoulli (Suc k)) * pochhammer (of-nat
```

k) *j* / (*of-nat k* ∗
of-nat (*Suc k*))) ∗ *inverse x* ˆ (*k* + *j*))

definition *stirling-sum′* **where**
stirling-sum′ j m x =
(−*1*) ˆ (*Suc j*) ∗ ($\sum$ *k*≤*m*. (*of-real* (*bernoulli′ k*) ∗
pochhammer (*of-nat* (*Suc k*)) (*j* − *1*) ∗ *inverse x* ˆ (*k* + *j*)))

lemma *stirling-sum-complex-of-real*:
stirling-sum j m (*complex-of-real x*) = *complex-of-real* (*stirling-sum j m x*)
by (*simp add*: *stirling-sum-def pochhammer-of-real* [*symmetric*] *del*: *of-nat-Suc*)

lemma *stirling-sum′-complex-of-real*:
stirling-sum′ j m (*complex-of-real x*) = *complex-of-real* (*stirling-sum′ j m x*)
by (*simp add*: *stirling-sum′-def pochhammer-of-real* [*symmetric*] *del*: *of-nat-Suc*)

lemma *has-field-derivative-stirling-sum-complex* [*derivative-intros*]:
Re x > *0* ⟹ (*stirling-sum j m has-field-derivative stirling-sum* (*Suc j*) *m x*) (*at x*)
unfolding *stirling-sum-def* [*abs-def*] *sum-distrib-left*
by (*rule DERIV-sum*) (*auto intro*!: *derivative-eq-intros simp del*: *of-nat-Suc*
simp: *pochhammer-Suc power-diff*)

lemma *has-field-derivative-stirling-sum-real* [*derivative-intros*]:
x > (*0*::*real*) ⟹ (*stirling-sum j m has-field-derivative stirling-sum* (*Suc j*) *m x*) (*at x*)
unfolding *stirling-sum-def* [*abs-def*] *sum-distrib-left*
by (*rule DERIV-sum*) (*auto intro*!: *derivative-eq-intros simp del*: *of-nat-Suc*
simp: *pochhammer-Suc power-diff*)

lemma *has-field-derivative-stirling-sum′-complex* [*derivative-intros*]:
assumes *j* > *0 Re x* > *0*
shows (*stirling-sum′ j m has-field-derivative stirling-sum′* (*Suc j*) *m x*) (*at x*)
proof (*cases j*)
case (*Suc j′*)
from *assms* **have** [*simp*]: *x* ≠ *0* **by** *auto*
define *c* **where** *c* = (λ*n*. (−*1*) ˆ *Suc j* ∗ *complex-of-real* (*bernoulli′ n*) ∗
pochhammer (*of-nat* (*Suc n*)) *j′*)
define *T* **where** *T* = (λ*n x*. *c n* ∗ *inverse x* ˆ (*j* + *n*))
define *T′* **where** *T′* = (λ*n x*. − (*of-nat* (*j* + *n*)) ∗ *c n* ∗ *inverse x* ˆ (*Suc* (*j* + *n*)))
have ((λ*x*. $\sum$ *k*≤*m*. *T k x*) *has-field-derivative* ($\sum$ *k*≤*m*. *T′ k x*)) (*at x*) **using** *assms Suc*
by (*intro DERIV-sum*)
(*auto simp*: *T-def T′-def intro*!: *derivative-eq-intros*
simp: *field-simps power-add* [*symmetric*] *simp del*: *of-nat-Suc power-Suc of-nat-add*)
also have (λ*x*. ($\sum$ *k*≤*m*. *T k x*)) = *stirling-sum′ j m*
by (*simp add*: *Suc T-def c-def stirling-sum′-def fun-eq-iff add-ac mult.assoc*

sum-distrib-left)
also have ($\sum k{\le}m.\ T'\ k\ x$) = *stirling-sum'* (*Suc j*) *m x*
by (*simp add*: *T'-def c-def Suc stirling-sum'-def sum-distrib-left*
sum-distrib-right algebra-simps pochhammer-Suc)
finally show *?thesis* **.**
qed (*insert assms*, *simp-all*)

lemma *has-field-derivative-stirling-sum'-real* [*derivative-intros*]:
assumes $j > 0$ $x > (0{::}real)$
shows (*stirling-sum' j m has-field-derivative stirling-sum'* (*Suc j*) *m x*) (*at x*)
proof (*cases j*)
case (*Suc j'*)
from *assms* **have** [*simp*]: $x \neq 0$ **by** *auto*
define *c* **where** $c = (\lambda n.\ (-1)$ ^ *Suc j* ∗ (*bernoulli' n*) ∗ *pochhammer* (*of-nat* (*Suc n*)) *j'*)
define *T* **where** $T = (\lambda n\ x.\ c\ n$ ∗ *inverse x* ^ $(j + n))$
define *T'* **where** $T' = (\lambda n\ x.\ -$ (*of-nat* $(j + n)$) ∗ *c n* ∗ *inverse x* ^ (*Suc* $(j + n)$))
have (($\lambda x.\ \sum k{\le}m.\ T\ k\ x$) *has-field-derivative* ($\sum k{\le}m.\ T'\ k\ x$)) (*at x*) **using** *assms Suc*
by (*intro DERIV-sum*)
(*auto simp*: *T-def T'-def intro*!: *derivative-eq-intros*
simp: *field-simps power-add* [*symmetric*] *simp del*: *of-nat-Suc power-Suc of-nat-add*)
also have ($\lambda x.$ ($\sum k{\le}m.\ T\ k\ x$)) = *stirling-sum' j m*
by (*simp add*: *Suc T-def c-def stirling-sum'-def fun-eq-iff add-ac mult.assoc sum-distrib-left*)
also have ($\sum k{\le}m.\ T'\ k\ x$) = *stirling-sum'* (*Suc j*) *m x*
by (*simp add*: *T'-def c-def Suc stirling-sum'-def sum-distrib-left*
sum-distrib-right algebra-simps pochhammer-Suc)
finally show *?thesis* **.**
qed (*insert assms*, *simp-all*)

lemma *higher-deriv-stirling-sum-complex*:
Re x > *0* ⟹ (*deriv* ^^ *i*) (*stirling-sum j m*) *x* = *stirling-sum* (*i* + *j*) *m x*
proof (*induction i arbitrary*: *x*)
case (*Suc i*)
have *deriv* ((*deriv* ^^ *i*) (*stirling-sum j m*)) *x* = *deriv* (*stirling-sum* (*i* + *j*) *m*) *x*
using *eventually-nhds-in-open*[*of* {*x*. *Re x* > *0*} *x*] *Suc.prems*
by (*intro deriv-cong-ev refl*) (*auto elim*!: *eventually-mono simp*: *open-halfspace-Re-gt Suc.IH*)
also from *Suc.prems* **have** ... = *stirling-sum* (*Suc* (*i* + *j*)) *m x*
by (*intro DERIV-imp-deriv has-field-derivative-stirling-sum-complex*)
finally show *?case* **by** *simp*
qed *simp-all*

definition *Polygamma-approx* :: *nat* ⇒ *nat* ⇒ *'a* ⇒ *'a* :: {*real-normed-field*, *ln*}

where
Polygamma-approx j m =
(*deriv* ^^ *j*) (λx. (*x* − *1* / *2*) ∗ *ln x* − *x* + *of-real* (*ln* (*2* ∗ *pi*)) / *2* + *stirling-sum 0 m x*)

lemma *Polygamma-approx-Suc*: *Polygamma-approx* (*Suc j*) *m* = *deriv* (*Polygamma-approx j m*)
by (*simp add*: *Polygamma-approx-def*)

lemma *Polygamma-approx-0*:
Polygamma-approx 0 m x = (*x* − *1*/*2*) ∗ *ln x* − *x* + *of-real* (*ln* (*2*∗*pi*)) / *2* + *stirling-sum 0 m x*
by (*simp add*: *Polygamma-approx-def*)

lemma *Polygamma-approx-1-complex*:
Re x > *0* ⟹
Polygamma-approx (*Suc 0*) *m x* = *ln x* − *1* / (*2*∗*x*) + *stirling-sum* (*Suc 0*) *m x*
unfolding *Polygamma-approx-Suc Polygamma-approx-0*
by (*intro DERIV-imp-deriv*)
(*auto intro*!: *derivative-eq-intros elim*!: *nonpos-Reals-cases simp*: *field-simps*)

lemma *Polygamma-approx-1-real*:
x > (*0* :: *real*) ⟹
Polygamma-approx (*Suc 0*) *m x* = *ln x* − *1* / (*2*∗*x*) + *stirling-sum* (*Suc 0*) *m x*
unfolding *Polygamma-approx-Suc Polygamma-approx-0*
by (*intro DERIV-imp-deriv*)
(*auto intro*!: *derivative-eq-intros elim*!: *nonpos-Reals-cases simp*: *field-simps*)

lemma *stirling-sum-2-conv-stirling-sum′-1*:
fixes *x* :: *′a* :: {*real-div-algebra*, *field-char-0*}
assumes *m* > *0 x* ≠ *0*
shows *stirling-sum′ 1 m x* = *1* / *x* + *1* / (*2* ∗ *x*^*2*) + *stirling-sum 2 m x*
proof −
have *pochhammer-2*: *pochhammer* (*of-nat k*) *2* = *of-nat k* ∗ *of-nat* (*Suc k*) **for** *k*
by (*simp add*: *pochhammer-Suc eval-nat-numeral add-ac*)
have *stirling-sum 2 m x* =
($\sum$ *k* = *Suc 0*..<*m*. *of-real* (*bernoulli′* (*Suc k*)) ∗ *inverse x* ^ *Suc* (*Suc k*))
unfolding *stirling-sum-def pochhammer-2 power2-minus power-one mult-1-left*
by (*intro sum.cong refl*)
(*simp-all add*: *stirling-sum-def pochhammer-2 power2-eq-square divide-simps bernoulli′-def*
del: *of-nat-Suc power-Suc*)
also have *1* / (*2* ∗ *x*^*2*) + ... =
($\sum$ *k*=*0*..<*m*. *of-real* (*bernoulli′* (*Suc k*)) ∗ *inverse x* ^ *Suc* (*Suc k*))
using *assms*
by (*subst* (*2*) *sum-head-upt-Suc*) (*simp-all add*: *power2-eq-square field-simps*)

also have *1 / x + . . .* = ($\sum$ *k=0..<Suc m. of-real* (*bernoulli′ k*) ∗ *inverse x* ˆ *Suc k*)
by (*subst sum.atLeast0-lessThan-Suc-shift*) (*simp-all add*: *bernoulli′-def divide-simps*)
also have . . . = ($\sum$ *k≤m. of-real* (*bernoulli′ k*) ∗ *inverse x* ˆ *Suc k*)
by (*intro sum.cong*) *auto*
also have . . . = *stirling-sum′ 1 m x* **by** (*simp add*: *stirling-sum′-def*)
finally show *?thesis* **by** (*simp add*: *add-ac*)
qed

lemma *Polygamma-approx-2-real*:
assumes *x* > (*0*::*real*) *m* > *0*
shows *Polygamma-approx* (*Suc* (*Suc 0*)) *m x* = *stirling-sum′ 1 m x*
proof −
have *Polygamma-approx* (*Suc* (*Suc 0*)) *m x* = *deriv* (*Polygamma-approx* (*Suc 0*) *m*) *x*
by (*simp add*: *Polygamma-approx-Suc*)
also have . . . = *deriv* (λ*x. ln x* − *1* / (*2*∗*x*) + *stirling-sum* (*Suc 0*) *m x*) *x*
using *eventually-nhds-in-open*[*of* {*0*<..} *x*] *assms*
by (*intro deriv-cong-ev*) (*auto elim*!: *eventually-mono simp*: *Polygamma-approx-1-real*)
also have . . . = *1* / *x* + *1* / (*2*∗*x*ˆ*2*) + *stirling-sum* (*Suc* (*Suc 0*)) *m x* **using** *assms*
by (*intro DERIV-imp-deriv*) (*auto intro*!: *derivative-eq-intros*
elim!: *nonpos-Reals-cases simp*: *field-simps power2-eq-square*)
also have . . . = *stirling-sum′ 1 m x* **using** *stirling-sum-2-conv-stirling-sum′-1*[*of m x*] *assms*
by (*simp add*: *eval-nat-numeral*)
finally show *?thesis* **.**
qed

lemma *Polygamma-approx-2-complex*:
assumes *Re x* > *0 m* > *0*
shows *Polygamma-approx* (*Suc* (*Suc 0*)) *m x* = *stirling-sum′ 1 m x*
proof −
have *Polygamma-approx* (*Suc* (*Suc 0*)) *m x* = *deriv* (*Polygamma-approx* (*Suc 0*) *m*) *x*
by (*simp add*: *Polygamma-approx-Suc*)
also have . . . = *deriv* (λ*x. ln x* − *1* / (*2*∗*x*) + *stirling-sum* (*Suc 0*) *m x*) *x*
using *eventually-nhds-in-open*[*of* {*s. Re s* > *0*} *x*] *assms*
by (*intro deriv-cong-ev*)
(*auto simp*: *open-halfspace-Re-gt elim*!: *eventually-mono simp*: *Polygamma-approx-1-complex*)
also have . . . = *1* / *x* + *1* / (*2*∗*x*ˆ*2*) + *stirling-sum* (*Suc* (*Suc 0*)) *m x* **using** *assms*
by (*intro DERIV-imp-deriv*) (*auto intro*!: *derivative-eq-intros*
elim!: *nonpos-Reals-cases simp*: *field-simps power2-eq-square*)
also have . . . = *stirling-sum′ 1 m x* **using** *stirling-sum-2-conv-stirling-sum′-1*[*of m x*] *assms*
by (*subst stirling-sum-2-conv-stirling-sum′-1*) (*auto simp*: *eval-nat-numeral*)
finally show *?thesis* **.**
qed

```
lemma Polygamma-approx-ge-2-real:
  assumes x > (0::real) m > 0
  shows   Polygamma-approx (Suc (Suc j)) m x = stirling-sum' (Suc j) m x
using assms(1)
proof (induction j arbitrary: x)
  case (0 x)
  with assms show ?case by (simp add: Polygamma-approx-2-real)
next
  case (Suc j x)
  have Polygamma-approx (Suc (Suc (Suc j))) m x = deriv (Polygamma-approx
(Suc (Suc j)) m) x
    by (simp add: Polygamma-approx-Suc)
  also have ... = deriv (stirling-sum' (Suc j) m) x
    using eventually-nhds-in-open[of {0<..} x] Suc.prems
    by (intro deriv-cong-ev refl) (auto elim!: eventually-mono simp: Suc.IH)
  also have ... = stirling-sum' (Suc (Suc j)) m x using Suc.prems
    by (intro DERIV-imp-deriv derivative-intros) simp-all
  finally show ?case .
qed

lemma Polygamma-approx-ge-2-complex:
  assumes Re x > 0 m > 0
  shows   Polygamma-approx (Suc (Suc j)) m x = stirling-sum' (Suc j) m x
using assms(1)
proof (induction j arbitrary: x)
  case (0 x)
  with assms show ?case by (simp add: Polygamma-approx-2-complex)
next
  case (Suc j x)
  have Polygamma-approx (Suc (Suc (Suc j))) m x = deriv (Polygamma-approx
(Suc (Suc j)) m) x
    by (simp add: Polygamma-approx-Suc)
  also have ... = deriv (stirling-sum' (Suc j) m) x
    using eventually-nhds-in-open[of {x. Re x > 0} x] Suc.prems
   by (intro deriv-cong-ev refl) (auto elim!: eventually-mono simp: Suc.IH open-halfspace-Re-gt)
  also have ... = stirling-sum' (Suc (Suc j)) m x using Suc.prems
    by (intro DERIV-imp-deriv derivative-intros) simp-all
  finally show ?case .
qed

lemma Polygamma-approx-complex-of-real:
  assumes x > 0 m > 0
  shows   Polygamma-approx j m (complex-of-real x) = of-real (Polygamma-approx
j m x)
proof (cases j)
  case 0
  with assms show ?thesis by (simp add: Polygamma-approx-0 Ln-of-real stirling-sum-complex-of-real)
next
```

case [*simp*]: (*Suc j'*)
thus *?thesis*
proof (*cases j'*)
case *0*
with *assms* **show** *?thesis*
by (*simp add*: *Polygamma-approx-1-complex*
Polygamma-approx-1-real stirling-sum-complex-of-real Ln-of-real)
next
case (*Suc j''*)
with *assms* **show** *?thesis*
by (*simp add*: *Polygamma-approx-ge-2-complex Polygamma-approx-ge-2-real*
stirling-sum'-complex-of-real)
qed
qed

lemma *higher-deriv-Polygamma-approx* [*simp*]:
(*deriv ^^ j*) (*Polygamma-approx i m*) = *Polygamma-approx* (*j* + *i*) *m*
by (*simp add*: *Polygamma-approx-def funpow-add*)

lemma *stirling-sum-holomorphic* [*holomorphic-intros*]:
0 ∉ *A* ⟹ *stirling-sum j m holomorphic-on A*
unfolding *stirling-sum-def* **by** (*intro holomorphic-intros*) *auto*

lemma *Polygamma-approx-holomorphic* [*holomorphic-intros*]:
Polygamma-approx j m holomorphic-on {*s*. *Re s* > *0*}
unfolding *Polygamma-approx-def*
by (*intro holomorphic-intros*) (*auto simp*: *open-halfspace-Re-gt elim*!: *nonpos-Reals-cases*)

lemma *higher-deriv-lnGamma-stirling*:
assumes *m*: *m* > *0*
shows (λ*x*::*real*. (*deriv ^^ j*) *ln-Gamma x* − *Polygamma-approx j m x*) ∈ *O*(λ*x*.
1 / *x* ^ (*m* + *j*))
proof −
have *eventually* (λ*x*. |(*deriv ^^ j*) *ln-Gamma x* − *Polygamma-approx j m x*| =
inverse (*real m*) ∗ |(*deriv ^^ j*) (*stirling-integral m*) *x*|) *at-top*
using *eventually-gt-at-top*[*of 0*::*real*]
proof *eventually-elim*
case (*elim x*)
note *x* = *this*
have (*deriv ^^ j*) (λ*x*. *ln-Gamma x* − *Polygamma-approx 0 m x*) (*complex-of-real*
x) =
(*deriv ^^ j*) (λ*x*. (−*inverse* (*of-nat m*)) ∗ *stirling-integral m x*)
(*complex-of-real x*)
using *eventually-nhds-in-open*[*of* {*s*. *Re s* > *0*} *x*] *x m*
by (*intro higher-deriv-cong-ev refl*)
(*auto elim*!: *eventually-mono simp*: *ln-Gamma-stirling-complex Polygamma-approx-def*

field-simps open-halfspace-Re-gt stirling-sum-def)
also have ... = − *inverse* (*of-nat m*) ∗ (*deriv ^^ j*) (*stirling-integral m*) (*of-real*

x) **using** *x m*
by (*intro higher-deriv-cmult*[*of* - {*s*. *Re s* > *0*}] *stirling-integral-holomorphic*)
(*auto simp*: *open-halfspace-Re-gt*)
also have (*deriv* ^^ *j*) (λ*x*. *ln-Gamma x* − *Polygamma-approx 0 m x*) (*complex-of-real* *x*) =
(*deriv* ^^ *j*) *ln-Gamma* (*of-real x*) − (*deriv* ^^ *j*) (*Polygamma-approx* *0 m*) (*of-real x*)
using *x*
by (*intro higher-deriv-diff* [*of* - {*s*. *Re s* > *0*}])
(*auto intro*!: *holomorphic-intros elim*!: *nonpos-Reals-cases simp*: *open-halfspace-Re-gt*)
also have (*deriv* ^^ *j*) (*Polygamma-approx 0 m*) (*complex-of-real x*) =
of-real (*Polygamma-approx j m x*) **using** *x m*
by (*simp add*: *Polygamma-approx-complex-of-real*)
also have *norm* (− *inverse* (*of-nat m*) ∗ (*deriv* ^^ *j*) (*stirling-integral m*) (*complex-of-real x*)) =
inverse (*real m*) ∗ |(*deriv* ^^ *j*) (*stirling-integral m*) *x*|
using *x m* **by** (*simp add*: *norm-mult norm-inverse deriv-stirling-integral-complex-of-real*)
also have (*deriv* ^^ *j*) *ln-Gamma* (*complex-of-real x*) = *of-real* ((*deriv* ^^ *j*) *ln-Gamma x*) **using** *x*
by (*simp add*: *higher-deriv-ln-Gamma-complex-of-real*)
also have *norm* (. . . − *of-real* (*Polygamma-approx j m x*)) =
|(*deriv* ^^ *j*) *ln-Gamma x* − *Polygamma-approx j m x*|
by (*simp only*: *of-real-diff* [*symmetric*] *norm-of-real*)
finally show *?case* **.**
qed
from *bigthetaI-cong*[*OF this*] *m*
have (λ*x*::*real*. (*deriv* ^^ *j*) *ln-Gamma x* − *Polygamma-approx j m x*) ∈
Θ(λ*x*. (*deriv* ^^ *j*) (*stirling-integral m*) *x*) **by** *simp*
also have (λ*x*::*real*. (*deriv* ^^ *j*) (*stirling-integral m*) *x*) ∈ *O*(λ*x*. *1* / *x* ^ (*m* + *j*)) **using** *m*
by (*rule deriv-stirling-integral-real-bound*)
finally show *?thesis* **.**
qed

lemma *Polygamma-approx-1-real′*:
assumes *x*: (*x*::*real*) > *0* **and** *m*: *m* > *0*
shows *Polygamma-approx 1 m x* = *ln x* − (∑*k* = *Suc 0*..*m*. *bernoulli′ k* ∗ *inverse x* ^ *k* / *real k*)
proof −
have *Polygamma-approx 1 m x* = *ln x* − (*1* / (*2* ∗ *x*) +
(∑*k*=*Suc 0*..<*m*. *bernoulli* (*Suc k*) ∗ *inverse x* ^ *Suc k* / *real* (*Suc k*)))
(**is** - = - − (- + *?S*)) **using** *x* **by** (*simp add*: *Polygamma-approx-1-real* *stirling-sum-def*)
also have *?S* = (∑*k*=*Suc 0*..<*m*. *bernoulli′* (*Suc k*) ∗ *inverse x* ^ *Suc k* / *real* (*Suc k*))
by (*intro sum.cong refl*) (*simp-all add*: *bernoulli′-def*)
also have *1* / (*2* ∗ *x*) + . . . =
(∑*k*=*0*..<*m*. *bernoulli′* (*Suc k*) ∗ *inverse x* ^ *Suc k* / *real* (*Suc k*))
using *m*

```
    by (subst (2) sum-head-upt-Suc) (simp-all add: field-simps)
  also have ... = (∑k = Suc 0..m. bernoulli' k * inverse x ^ k / real k) using
assms
   by (subst sum-shift-bounds-Suc-ivl [symmetric]) (simp add: atLeastLessThanSuc-atLeastAtMost)
  finally show ?thesis .
qed

theorem
  assumes m: m > 0
  shows   ln-Gamma-real-asymptotics:
          (λx. ln-Gamma x − ((x − 1 / 2) * ln x − x + ln (2 * pi) / 2 +
               (∑k = 1..<m. bernoulli (Suc k) / (real k * real (Suc k)) / x^k)))
            ∈ O(λx. 1 / x ^ m) (is ?th1)
    and   Digamma-real-asymptotics:
          (λx. Digamma x − (ln x − (∑k=1..m. bernoulli' k / real k / x ^ k)))
            ∈ O(λx. 1 / (x ^ Suc m)) (is ?th2)
    and   Polygamma-real-asymptotics: j > 0 ⟹
           (λx. Polygamma j x − (− 1) ^ Suc j * (∑k≤m. bernoulli' k *
                pochhammer (real (Suc k)) (j − 1) / x ^ (k + j)))
            ∈ O(λx. 1 / x ^ (m+j+1)) (is - ⟹ ?th3)
proof −
  define G :: nat ⇒ real ⇒ real where
    G = (λm. if m = 0 then ln-Gamma else Polygamma (m − 1))
  have *: (λx. G j x − h x) ∈ O(λx. 1 / x ^ (m + j))
    if ⋀x::real. x > 0 ⟹ Polygamma-approx j m x = h x for j h
  proof −
    have (λx. G j x − h x) ∈
              Θ(λx. (deriv ^^ j) ln-Gamma x − Polygamma-approx j m x) (is - ∈
Θ(?f))
      using that
     by (intro bigthetaI-cong) (auto intro: eventually-mono[OF eventually-gt-at-top[of
0::real]]
          simp del: funpow.simps simp: higher-deriv-ln-Gamma-real G-def)
    also have ?f ∈ O(λx::real. 1 / x ^ (m + j)) using m
      by (rule higher-deriv-lnGamma-stirling)
    finally show ?thesis .
  qed

  note [[simproc del: simplify-landau-sum]]
  from *[OF Polygamma-approx-0] assms show ?th1
    by (simp add: G-def Polygamma-approx-0 stirling-sum-def field-simps)
  from *[OF Polygamma-approx-1-real'] assms show ?th2 by (simp add: G-def
field-simps)

  assume j: j > 0
  from *[OF Polygamma-approx-ge-2-real, of j − 1] assms j show ?th3
    by (simp add: G-def stirling-sum'-def power-add power-diff field-simps)
qed
```

end

References

[1] B. Berndt. *Rudiments of the Theory of the Gamma Function.* University of Chicago, 1976.

[2] A. Erdélyi. *Asymptotic Expansions.* Dover Publications, 1956.

[3] G. J. O. Jameson. A simple proof of Stirling's formula for the Gamma function. *The Mathematical Gazette*, 99:68–74, 3 2015.