

Prime Number Theorem with Remainder Term

Shuhao Song and Bowen Yao

June 20, 2024

Abstract

We have formalized the proof of the Prime Number Theorem with remainder term. This is the first formalized version of PNT with an explicit error term.

There are many useful results in this AFP entry.

First, the main result, prime number theorem with remainder:

$$\pi(x) = \text{Li}(x) + O\left(x \exp\left(-\sqrt{\log x/3653}\right)\right)$$

Second, the zero-free region of the Riemann zeta function:

$$\zeta(\beta + i\gamma) \neq 0 \text{ when } \beta \geq 1 - \frac{1}{952320} (\log(|\gamma| + 2))^{-1}$$

Moreover, we proved a revised version of Perron's formula, together with the zero-free region we can prove the main result.

Contents

1	Auxiliary library for prime number theorem	2
1.1	Zeta function	2
1.2	Logarithm derivatives	3
1.3	Lemmas of integration and integrability	6
1.4	Lemmas on asymptotics	10
1.5	Lemmas of <i>floor</i> , <i>ceil</i> and <i>nat_pow</i>	11
1.6	Elementary estimation of <i>exp</i> and <i>ln</i>	12
1.7	Miscellaneous lemmas	13
2	Implication relation of many forms of prime number theorem	14
3	Some basic theorems in complex analysis	21
3.1	Introduction rules for holomorphic functions and analytic functions	21
3.2	Factorization of analytic function on compact region	22
3.2.1	Auxiliary propositions for theorem <i>analytic_factorization</i>	24
3.3	Schwarz theorem in complex analysis	27
3.4	Borel-Carathedory theorem	28
3.5	Lemma 3.9	30
4	Zero-free region of zeta function	36
5	Perron's formula	58
6	Estimation of the order of $\frac{\zeta'(s)}{\zeta(s)}$	77

7 Deducing prime number theorem using Perron's formula

```

theory PNT_Notation
imports
  Prime_Number_Theorem.Prime_Counting_Functions
begin

definition PNT_const_C1  $\equiv 1 / 952320 :: real$ 

abbreviation nat_powr
  (infixr nat'_powr 80)
where
  n nat_powr x  $\equiv (of\_nat n) powr x$ 

bundle pnt_notation
begin
notation PNT_const_C1 (C1)
notation norm (||_||)
notation Suc ( $\_+ [101] 100$ )
end

bundle no_pnt_notation
begin
no_notation PNT_const_C1 (C1)
no_notation norm (||_||)
no_notation Suc ( $\_+ [101] 100$ )
end

end
theory PNT_Remainder_Library
imports
  PNT_Notation
begin
unbundle pnt_notation

```

1 Auxiliary library for prime number theorem

1.1 Zeta function

```

lemma pre_zeta_1_bound:
  assumes  $0 < Re\ s$ 
  shows  $\|pre\_zeta\ 1\ s\| \leq \|s\| / Re\ s$ 
proof -
  have  $\|pre\_zeta\ 1\ s\| \leq \|s\| / (Re\ s * 1\ powr\ Re\ s)$ 
    by (rule pre_zeta_bound') (use assms in auto)
  also have  $\dots = \|s\| / Re\ s$  by auto
  finally show ?thesis .
qed

lemma zeta_pole_eq:
  assumes  $s \neq 1$ 
  shows  $zeta\ s = pre\_zeta\ 1\ s + 1 / (s - 1)$ 
proof -
  have  $zeta\ s - 1 / (s - 1) = pre\_zeta\ 1\ s$  by (intro zeta_minus_pole_eq assms)
  thus ?thesis by (simp add: field_simps)
qed

```

definition $zeta'$ where $zeta' s \equiv pre_zeta 1 s * (s - 1) + 1$

lemma $zeta'_analytic$:

$zeta'$ analytic_on UNIV

unfolding $zeta'_def$ by (intro analytic_intros) auto

lemma $zeta'_analytic_on$ [analytic_intros]:

$zeta'$ analytic_on A using $zeta'_analytic$ analytic_on_subset by auto

lemma $zeta'_holomorphic_on$ [holomorphic_intros]:

$zeta'$ holomorphic_on A using $zeta'_analytic_on$ by (intro analytic_imp_holomorphic)

lemma $zeta_eq_zeta'$:

$zeta s = zeta' s / (s - 1)$

proof (cases $s = 1$)

case True thus ?thesis using $zeta_1$ unfolding $zeta'_def$ by auto

next

case False with $zeta_pole_eq$ [OF this]

show ?thesis unfolding $zeta'_def$ by (auto simp add: field_simps)

qed

lemma $zeta'_1$ [simp]: $zeta' 1 = 1$ unfolding $zeta'_def$ by auto

lemma $zeta_eq_zero_iff_zeta'$:

shows $s \neq 1 \implies zeta' s = 0 \iff zeta s = 0$

using $zeta_eq_zeta'$ [of s] by auto

lemma $zeta'_eq_zero_iff$:

shows $zeta' s = 0 \iff zeta s = 0 \wedge s \neq 1$

by (cases $s = 1$, use $zeta_eq_zero_iff_zeta'$ in auto)

lemma $zeta_eq_zero_iff$:

shows $zeta s = 0 \iff zeta' s = 0 \vee s = 1$

by (subst $zeta'_eq_zero_iff$, use $zeta_1$ in auto)

1.2 Logarithm derivatives

definition $logderiv f x \equiv deriv f x / f x$

definition $log_differentiable$

(infixr ($log'_differentiable$) 50)

where

$f log_differentiable x \equiv (f field_differentiable (at x)) \wedge f x \neq 0$

lemma $logderiv_prod'$:

fixes $f :: 'n \Rightarrow 'f \Rightarrow 'f :: real_normed_field$

assumes fin : finite I

and $lder$: $\bigwedge i. i \in I \implies f i log_differentiable a$

shows $logderiv (\lambda x. \prod_{i \in I}. f i x) a = (\sum_{i \in I}. logderiv (f i) a)$ (is ?P)

and $(\lambda x. \prod_{i \in I}. f i x) log_differentiable a$ (is ?Q)

proof –

let ?a = $\lambda i. deriv (f i) a$

let ?b = $\lambda i. \prod_{j \in I - \{i\}}. f j a$

let ?c = $\lambda i. f i a$

let ?d = $\prod_{i \in I}. ?c i$

have der : $\bigwedge i. i \in I \implies f i field_differentiable (at a)$

and $\text{nz}: \bigwedge i. i \in I \implies f i a \neq 0$
using `lder unfolding log_differentiable_def` **by** `auto`
have $1: (*) x = (\lambda y. y * x)$ **for** $x :: 'f$ **by** `auto`
have $((\lambda x. \prod i \in I. f i x)$ *has_derivative*
 $(\lambda y. \sum i \in I. ?a i * y * ?b i))$ *(at a within UNIV)*
by $(\text{rule } \text{has_derivative_prod, fold has_field_derivative_def})$
 $(\text{rule } \text{field_differentiable_derivI, elim der})$
hence $2: \text{DERIV } (\lambda x. \prod i \in I. f i x) a :> (\sum i \in I. ?a i * ?b i)$
unfolding `has_field_derivative_def`
by $(\text{simp add: sum_distrib_left [symmetric] mult_ac})$
 $(\text{subst } 1, \text{blast})$
have $\text{prod_nz}: (\prod i \in I. ?c i) \neq 0$
using `prod_zero_iff nz fin` **by** `auto`
have $\text{mult_cong}: b = c \implies a * b = a * c$ **for** $a b c :: \text{real}$ **by** `auto`
have $\text{logderiv } (\lambda x. \prod i \in I. f i x) a = \text{deriv } (\lambda x. \prod i \in I. f i x) a / ?d$
unfolding `logderiv_def` **by** `auto`
also have $\dots = (\sum i \in I. ?a i * ?b i) / ?d$
using $2 \text{ DERIV_imp_deriv}$ **by** `auto`
also have $\dots = (\sum i \in I. ?a i * (?b i / ?d))$
by $(\text{auto simp add: sum_divide_distrib})$
also have $\dots = (\sum i \in I. \text{logderiv } (f i) a)$
proof –
have $\bigwedge a b c :: 'f. a \neq 0 \implies a = b * c \implies c / a = \text{inverse } b$
by $(\text{auto simp add: field_simps})$
moreover have $?d = ?c i * ?b i$ **if** $i \in I$ **for** i
by $(\text{intro prod.remove that fin})$
ultimately have $?b i / ?d = \text{inverse } (?c i)$ **if** $i \in I$ **for** i
using `prod_nz that` **by** `auto`
thus *?thesis* **unfolding** `logderiv_def` **using** 2
by $(\text{auto simp add: divide_inverse intro: sum.cong})$
qed
finally show $?P$.
show $?Q$ **by** $(\text{auto}$
 $\text{simp: log_differentiable_def field_differentiable_def}$
 $\text{intro!: } 2 \text{ prod_nz})$
qed

lemma `logderiv_prod`:
fixes $f :: 'n \Rightarrow 'f \Rightarrow 'f :: \text{real_normed_field}$
assumes `lder`: $\bigwedge i. i \in I \implies f i \text{ log_differentiable } a$
shows $\text{logderiv } (\lambda x. \prod i \in I. f i x) a = (\sum i \in I. \text{logderiv } (f i) a)$ **(is** $?P)$
and $(\lambda x. \prod i \in I. f i x) \text{ log_differentiable } a$ **(is** $?Q)$
proof –
consider $\text{finite } I \mid \text{infinite } I$ **by** `auto`
hence $?P \wedge ?Q$
proof *cases*
assume $\text{fin}: \text{finite } I$
show *?thesis* **by** $(\text{auto intro: logderiv_prod' lder fin})$
next
assume $\text{nfin}: \text{infinite } I$
show *?thesis* **using** nfin
unfolding `logderiv_def log_differentiable_def` **by** `auto`
qed
thus $?P ?Q$ **by** `auto`
qed

lemma *logderiv_mult*:
assumes f *log_differentiable* a
and g *log_differentiable* a
shows $\text{logderiv } (\lambda z. f z * g z) a = \text{logderiv } f a + \text{logderiv } g a$ (**is** ? P)
and $(\lambda z. f z * g z)$ *log_differentiable* a (**is** ? Q)

proof –
have $\text{logderiv } (\lambda z. f z * g z) a$
 $= \text{logderiv } (\lambda z. \prod_{i \in \{0, 1\}} ([f, g]!i) z) a$ **by** *auto*
also have $\dots = (\sum_{i \in \{0, 1\}} \text{logderiv } ([f, g]!i) a)$
by (*rule logderiv_prod(1)*, *use assms in auto*)
also have $\dots = \text{logderiv } f a + \text{logderiv } g a$
by *auto*
finally show ? P .
have $(\lambda z. \prod_{i \in \{0, 1\}} ([f, g]!i) z)$ *log_differentiable* a
by (*rule logderiv_prod(2)*, *use assms in auto*)
thus ? Q **by** *auto*

qed

lemma *logderiv_cong_ev*:
assumes $\forall_F x$ *in nhds* x . $f x = g x$
and $x = y$
shows $\text{logderiv } f x = \text{logderiv } g y$

proof –
have $\text{deriv } f x = \text{deriv } g y$ **using** *assms* **by** (*rule deriv_cong_ev*)
moreover have $f x = g y$ **using** *assms* **by** (*auto intro: eventually_nhds_x_imp_x*)
ultimately show ?*thesis* **unfolding** *logderiv_def* **by** *auto*

qed

lemma *logderiv_linear*:
assumes $z \neq a$
shows $\text{logderiv } (\lambda w. w - a) z = 1 / (z - a)$
and $(\lambda w. w - z)$ *log_differentiable* a
unfolding *logderiv_def log_differentiable_def*
using *assms* **by** (*auto simp add: derivative_intros*)

lemma *deriv_shift*:
assumes f *field_differentiable at* $(a + x)$
shows $\text{deriv } (\lambda t. f (a + t)) x = \text{deriv } f (a + x)$
proof –
have $\text{deriv } (f \circ (\lambda t. a + t)) x = \text{deriv } f (a + x)$
by (*subst deriv_chain*) (*auto intro: assms*)
thus ?*thesis* **unfolding** *comp_def* **by** *auto*

qed

lemma *logderiv_shift*:
assumes f *field_differentiable at* $(a + x)$
shows $\text{logderiv } (\lambda t. f (a + t)) x = \text{logderiv } f (a + x)$
unfolding *logderiv_def* **by** (*subst deriv_shift*) (*auto intro: assms*)

lemma *logderiv_inverse*:
assumes $x \neq 0$
shows $\text{logderiv } (\lambda x. 1 / x) x = - 1 / x$

proof –
have $\text{deriv } (\lambda x. 1 / x) x = (\text{deriv } (\lambda x. 1) x * x - 1 * \text{deriv } (\lambda x. x) x) / x^2$

by (rule deriv_divide) (use assms in auto)
 hence deriv $(\lambda x. 1 / x) x = - 1 / x^2$ by auto
 thus ?thesis unfolding logderiv_def power2_eq_square using assms by auto
 qed

lemma logderiv_zeta_eq_zeta':

assumes $s \neq 1$ zeta $s \neq 0$
 shows logderiv zeta $s = \text{logderiv zeta}' s - 1 / (s - 1)$

proof -

have logderiv zeta $s = \text{logderiv } (\lambda s. \text{zeta}' s * (1 / (s - 1))) s$
 using zeta_eq_zeta' by auto metis
 also have ... = logderiv zeta' $s + \text{logderiv } (\lambda s. 1 / (s - 1)) s$

proof -

have zeta' $s \neq 0$ using assms zeta_eq_zero_iff_zeta' by auto

hence zeta' log_differentiable s

unfolding log_differentiable_def

by (intro conjI analytic_on_imp_differentiable_at)
 (rule zeta'_analytic, auto)

moreover have $(\lambda z. 1 / (z - 1))$ log_differentiable s

unfolding log_differentiable_def using assms(1)

by (intro derivative_intros conjI, auto)

ultimately show ?thesis using assms by (intro logderiv_mult(1))

qed

also have logderiv $(\lambda s. 1 / (- 1 + s)) s = \text{logderiv } (\lambda s. 1 / s) (- 1 + s)$

by (rule logderiv_shift) (insert assms(1), auto intro: derivative_intros)

moreover have ... = $- 1 / (- 1 + s)$

by (rule logderiv_inverse) (use assms(1) in auto)

ultimately show ?thesis by auto

qed

lemma analytic_logderiv [analytic_intros]:

assumes f analytic_on $A \wedge z. z \in A \implies f z \neq 0$

shows $(\lambda s. \text{logderiv } f s)$ analytic_on A

using assms unfolding logderiv_def by (intro analytic_intros)

1.3 Lemmas of integration and integrability

lemma powr_has_integral:

fixes $a b w :: \text{real}$

assumes $Hab: a \leq b$ and $Hw: w > 0 \wedge w \neq 1$

shows $((\lambda x. w \text{ powr } x)$ has_integral $w \text{ powr } b / \ln w - w \text{ powr } a / \ln w$) $\{a..b\}$

proof (rule fundamental_theorem_of_calculus)

show $a \leq b$ using assms by auto

next

fix x assume $x \in \{a..b\}$

have $((\lambda x. \text{exp } (x * \ln w))$ has_vector_derivative $\text{exp } (x * \ln w) * (1 * \ln w)$) (at x within $\{a..b\}$)

by (subst has_real_derivative_iff_has_vector_derivative [symmetric])

(rule derivative_intros DERIV_cmult_right)+

hence $((\text{powr}) w$ has_vector_derivative $w \text{ powr } x * \ln w$) (at x within $\{a..b\}$)

unfolding powr_def using Hw by (simp add: DERIV_fun_exp)

moreover have $\ln w \neq 0$ using Hw by auto

ultimately show $((\lambda x. w \text{ powr } x / \ln w)$ has_vector_derivative $w \text{ powr } x$) (at x within $\{a..b\}$)

by (auto intro: derivative_eq_intros)

qed

lemma powr_integrable:

fixes $a\ b\ w :: \text{real}$
assumes $Hab: a \leq b$ **and** $Hw: w > 0 \wedge w \neq 1$
shows $(\lambda x. w \text{ powr } x) \text{ integrable_on } \{a..b\}$
by $(\text{rule } \text{has_integral_integrable}, \text{rule } \text{powr_has_integral})$
 $(\text{use } \text{assms } \text{in } \text{auto})$

lemma $\text{powr_integral_bound_gt_1}$:

fixes $a\ b\ w :: \text{real}$
assumes $Hab: a \leq b$ **and** $Hw: w > 1$
shows $\text{integral } \{a..b\} (\lambda x. w \text{ powr } x) \leq w \text{ powr } b / |\ln w|$
proof –
have $\text{integral } \{a..b\} (\lambda x. w \text{ powr } x) = w \text{ powr } b / \ln w - w \text{ powr } a / \ln w$
by $(\text{intro } \text{integral_unique_powr_has_integral}) (\text{use } \text{assms } \text{in } \text{auto})$
also have $\dots \leq w \text{ powr } b / |\ln w|$ **using** Hw **by** auto
finally show $?thesis$.

qed

lemma $\text{powr_integral_bound_lt_1}$:

fixes $a\ b\ w :: \text{real}$
assumes $Hab: a \leq b$ **and** $Hw: 0 < w \wedge w < 1$
shows $\text{integral } \{a..b\} (\lambda x. w \text{ powr } x) \leq w \text{ powr } a / |\ln w|$
proof –
have $\text{integral } \{a..b\} (\lambda x. w \text{ powr } x) = w \text{ powr } b / \ln w - w \text{ powr } a / \ln w$
by $(\text{intro } \text{integral_unique_powr_has_integral}) (\text{use } \text{assms } \text{in } \text{auto})$
also have $\dots \leq w \text{ powr } a / |\ln w|$ **using** Hw **by** $(\text{auto } \text{simp } \text{add: } \text{field_simps})$
finally show $?thesis$.

qed

lemma $\text{set_integrableI_bounded}$:

fixes $f :: 'a \Rightarrow 'b :: \{\text{banach}, \text{second_countable_topology}\}$
shows $A \in \text{sets } M$
 $\implies (\lambda x. \text{indicator } A\ x *_{\mathbb{R}} f\ x) \in \text{borel_measurable } M$
 $\implies \text{emeasure } M\ A < \infty$
 $\implies (AE\ x \text{ in } M. x \in A \implies \text{norm } (f\ x) \leq B)$
 $\implies \text{set_integrable } M\ A\ f$

unfolding $\text{set_integrable_def}$

by $(\text{rule } \text{integrableI_bounded_set}[\text{where } A=A]) \text{ auto}$

lemma $\text{integrable_cut}'$:

fixes $a\ b\ c :: \text{real}$ **and** $f :: \text{real} \Rightarrow \text{real}$
assumes $a \leq b \leq c$
and $Hf: \bigwedge x. a \leq x \implies f \text{ integrable_on } \{a..x\}$
shows $f \text{ integrable_on } \{b..c\}$

proof –

have $a \leq c$ **using** assms **by** linarith
hence $f \text{ integrable_on } \{a..c\}$ **by** $(\text{rule } Hf)$
thus $?thesis$ **by**
 $(\text{rule } \text{integrable_subinterval_real})$
 $(\text{subst } \text{subset_iff}, (\text{subst } \text{atLeastAtMost_iff})+)$,
 $\text{blast } \text{intro: } \langle a \leq b \rangle \text{ order_trans } [\text{of } a\ b]$

qed

lemma $\text{integration_by_part}'$:

fixes $a\ b :: \text{real}$
and $f\ g :: \text{real} \Rightarrow 'a :: \{\text{real_normed_field}, \text{banach}\}$

and $f' g' :: \text{real} \Rightarrow 'a$
assumes $a \leq b$
and $\bigwedge x. x \in \{a..b\} \implies (f \text{ has_vector_derivative } f' x) \text{ (at } x)$
and $\bigwedge x. x \in \{a..b\} \implies (g \text{ has_vector_derivative } g' x) \text{ (at } x)$
and $\text{int}: (\lambda x. f x * g' x) \text{ integrable_on } \{a..b\}$
shows $((\lambda x. f' x * g x) \text{ has_integral } f b * g b - f a * g a - \text{integral}\{a..b\} (\lambda x. f x * g' x)) \{a..b\}$
proof –
define prod **where** $\text{prod} \equiv (*) :: 'a \Rightarrow 'a \Rightarrow 'a$
define y **where** $y \equiv f b * g b - f a * g a - \text{integral}\{a..b\} (\lambda x. f x * g' x)$
have 0 : $\text{bounded_bilinear prod}$ **unfolding** prod_def
by $(\text{rule bounded_bilinear_mult})$
have 1 : $((\lambda x. f x * g' x) \text{ has_integral } f b * g b - f a * g a - y) \{a..b\}$
using y_def **and** int **and** $\text{integrable_integral}$ **by** auto
note $2 = \text{integration_by_parts}$
 $[\text{where } y = y \text{ and } \text{prod} = \text{prod}, \text{OF } 0, \text{unfolded } \text{prod_def}]$
have $\text{continuous_on } \{a..b\} f \text{ continuous_on } \{a..b\} g$
by $(\text{auto intro: has_vector_derivative_continuous } \text{has_vector_derivative_at_within } \text{assms } \text{simp: continuous_on_eq_continuous_within})$
with assms **and** 1 **show** $?thesis$ **by** $(\text{fold } y_def, \text{intro } 2) \text{ auto}$
qed

lemma integral_bigo :

fixes $a :: \text{real}$ **and** $f g :: \text{real} \Rightarrow \text{real}$
assumes $f_bound: f \in O(g)$
and $Hf: \bigwedge x. a \leq x \implies f \text{ integrable_on } \{a..x\}$
and $Hf': \bigwedge x. a \leq x \implies (\lambda x. |f x|) \text{ integrable_on } \{a..x\}$
and $Hg': \bigwedge x. a \leq x \implies (\lambda x. |g x|) \text{ integrable_on } \{a..x\}$
shows $(\lambda x. \text{integral}\{a..x\} f) \in O(\lambda x. 1 + \text{integral}\{a..x\} (\lambda x. |g x|))$
proof –
from $\langle f \in O(g) \rangle$ **obtain** c **where** $\forall_F x \text{ in } \text{at_top}. |f x| \leq c * |g x|$
unfolding bigo_def **by** auto
then **obtain** $N' :: \text{real}$ **where** $\text{asympt: } \bigwedge n. n \geq N' \implies |f n| \leq c * |g n|$
by $(\text{subst } (\text{asm}) \text{eventually_at_top_linorder}) (\text{blast})$
define N **where** $N \equiv \max a N'$
define I **where** $I \equiv |\text{integral } \{a..N\} f|$
define J **where** $J \equiv \text{integral } \{a..N\} (\lambda x. |g x|)$
define c' **where** $c' \equiv \max (I + J * |c|) |c|$
have $\bigwedge x. N \leq x \implies |\text{integral } \{a..x\} f| \leq c' * |1 + \text{integral } \{a..x\} (\lambda x. |g x|)|$
proof –
fix $x :: \text{real}$
assume $1: N \leq x$
define K **where** $K \equiv \text{integral } \{a..x\} (\lambda x. |g x|)$
have $2: a \leq N$ **unfolding** N_def **by** linarith
hence $3: a \leq x$ **using** 1 **by** linarith
have $\text{nnegs}: 0 \leq I \ 0 \leq J \ 0 \leq K$
unfolding $I_def J_def K_def$ **using** $1 \ 2 \ Hg'$
by $(\text{auto intro!: integral_nonneg})$
hence $\text{abs_eq}: |I| = I \ |J| = J \ |K| = K$
using nnegs **by** simp+
have $\text{int}|f|: (\lambda x. |f x|) \text{ integrable_on } \{N..x\}$
using $2 \ 1 \ Hf'$ **by** $(\text{rule integrable_cut'})$
have $\text{int}f: f \text{ integrable_on } \{N..x\}$


```

  using 2 1 Hf by (rule integrable_cut')
have  $\bigwedge x. a \leq x \implies (\lambda x. c * |g x|)$  integrable_on {a..x}
  by (blast intro: Hg' integrable_cmul [OF Hg', simplified])
hence  $\text{intc}|g|$ :  $(\lambda x. c * |g x|)$  integrable_on {N..x}
  using 2 1 by (blast intro: integrable_cut')
have  $|\text{integral } \{a..x\} f| \leq I + |\text{integral } \{N..x\} f|$ 
  unfolding I_def
  by (subst Henstock_Kurzweil_Integration.integral_combine
      [OF 2 1 Hf [of x], THEN sym])
      (rule 3, rule abs_triangle_ineq)
also have  $\dots \leq I + \text{integral } \{N..x\} (\lambda x. |f x|)$ 
proof -
  note integral_norm_bound_integral [OF intf int|f|]
  then have  $|\text{integral } \{N..x\} f| \leq \text{integral } \{N..x\} (\lambda x. |f x|)$  by auto
  then show ?thesis by linarith
qed
also have  $\dots \leq I + c * \text{integral } \{N..x\} (\lambda x. |g x|)$ 
proof -
  have 1:  $N' \leq N$  unfolding N_def by linarith
  hence  $\bigwedge y :: \text{real}. N \leq y \implies |f y| \leq c * |g y|$ 
  proof -
    fix y :: real
    assume  $N \leq y$ 
    thus  $|f y| \leq c * |g y|$ 
      by (rule asymp [OF order_trans [OF 1]])
  qed
  hence  $\text{integral } \{N..x\} (\lambda x. |f x|) \leq \text{integral } \{N..x\} (\lambda x. c * |g x|)$ 
    by (rule integral_le [OF intf |f| intc|g|]) simp
  thus ?thesis by simp
qed
also have  $\dots \leq I + |c| * (J + \text{integral } \{a..x\} (\lambda x. \|g x\|))$ 
proof -
  note Henstock_Kurzweil_Integration.integral_combine [OF 2 1 Hg' [of x]]
  hence  $K\_min\_J$ :  $\text{integral } \{N..x\} (\lambda x. |g x|) = K - J$ 
    unfolding J_def K_def using 3 by auto
  have  $c * (K - J) \leq |c| * (J + K)$  proof -
    have  $c * (K - J) \leq |c * (K - J)|$  by simp
    also have  $\dots = |c| * |K - J|$  by (simp add: abs_mult)
    also have  $\dots \leq |c| * (|J| + |K|)$  by (simp add: mult_left_mono)
    finally show ?thesis by (simp add: abs_eq)
  qed
  thus ?thesis by simp (subst K_min_J, fold K_def)
qed
also have  $\dots = (I + J * |c|) + |c| * \text{integral } \{a..x\} (\lambda x. |g x|)$ 
  by (simp add: field_simps)
also have  $\dots \leq c' + c' * \text{integral } \{a..x\} (\lambda x. |g x|)$ 
proof -
  have  $I + J * |c| \leq c'$  unfolding c'_def by auto
  thus ?thesis unfolding c'_def
    by (auto intro!: add_mono mult_mono integral_nonneg Hg' 3)
qed
finally show  $|\text{integral } \{a..x\} f|$ 
   $\leq c' * |1 + \text{integral } \{a..x\} (\lambda x. |g x|)|$ 
  by (simp add: integral_nonneg Hg' 3 field_simps)
qed

```

note $0 = \text{this}$
show *?thesis proof* (rule eventually_mono [THEN bigoI])
show $\forall_F x$ in at_top. $N \leq x$ **by** simp
show $\bigwedge x. N \leq x \implies \|\text{integral } \{a..x\} f\| \leq c' * \|\mathbf{1} + \text{integral } \{a..x\} (\lambda x. |g x|)\|$ **by** (simp, rule 0)
qed
qed

lemma integral_linepath_same_Re:
assumes Ha: $\text{Re } a = \text{Re } b$
and Hb: $\text{Im } a < \text{Im } b$
and Hf: (f has_contour_integral x) (linepath a b)
shows $((\lambda t. f (\text{Complex } (\text{Re } a) t) * i) \text{ has_integral } x) \{\text{Im } a.. \text{Im } b\}$

proof –
define path **where** path \equiv linepath a b
define c d e g **where** c \equiv Re a **and** d \equiv Im a **and** e \equiv Im b **and** g \equiv e – d
hence [simp]: $a = \text{Complex } c d$ $b = \text{Complex } c e$ **by** auto (subst Ha, auto)
have hg: $0 < g$ **unfolding** g_def **using** Hb **by** auto
have [simp]: $a *_R z = a * z$ **for** a **and** z :: complex **by** (rule complex_eqI) auto
have $((\lambda t. f (\text{path } t) * (b - a)) \text{ has_integral } x) \{0..1\}$
unfolding path_def **by** (subst has_contour_integral_linepath [symmetric]) (intro Hf)
moreover **have** path t = Complex c (g *_R t + d) **for** t
unfolding path_def linepath_def g_def
by (auto simp add: field_simps legacy_Complex_simps)
moreover **have** b – a = g * i
unfolding g_def **by** (auto simp add: legacy_Complex_simps)
ultimately **have**
 $((\lambda t. f (\text{Complex } c (g *_R t + d)) * (g * i)) \text{ has_integral } g * x /_R g \wedge \text{DIM}(\text{real}))$
 $(\text{cbox } ((d - d) /_R g) ((e - d) /_R g))$
by (subst (6) g_def) (auto simp add: field_simps)
hence $((\lambda t. f (\text{Complex } c t) * i * g) \text{ has_integral } x * g) \{d..e\}$
by (subst (asm) has_integral_affinity_iff)
 $(\text{auto simp add: field_simps hg})$
hence $((\lambda t. f (\text{Complex } c t) * i * g * (1 / g)) \text{ has_integral } x * g * (1 / g)) \{d..e\}$
by (rule has_integral_mult_left)
thus *?thesis* **using** hg **by** auto
qed

1.4 Lemmas on asymptotics

lemma eventually_at_top_linorderI':
fixes c :: 'a :: {no_top, linorder}
assumes h: $\bigwedge x. c < x \implies P x$
shows eventually P at_top
proof (rule eventually_mono)
show $\forall_F x$ in at_top. $c < x$ **by** (rule eventually_gt_at_top)
from h **show** $\bigwedge x. c < x \implies P x$.
qed

lemma eventually_le_imp_bigo:
assumes $\forall_F x$ in F. $\|f x\| \leq g x$
shows $f \in O[F](g)$
proof –
from assms **have** $\forall_F x$ in F. $\|f x\| \leq 1 * \|g x\|$ **by** eventually_elim auto
thus *?thesis* **by** (rule bigoI)
qed

lemma *eventually_le_imp_bigo'*:

assumes $\forall_F x \text{ in } F. \|f x\| \leq g x$

shows $(\lambda x. \|f x\|) \in O[F](g)$

proof –

from *assms* **have** $\forall_F x \text{ in } F. \|\|f x\|\| \leq 1 * \|g x\|$

by *eventually_elim auto*

thus *?thesis* **by** (*rule bigoI*)

qed

lemma *le_imp_bigo*:

assumes $\bigwedge x. \|f x\| \leq g x$

shows $f \in O[F](g)$

by (*intro eventually_le_imp_bigo eventuallyI assms*)

lemma *le_imp_bigo'*:

assumes $\bigwedge x. \|f x\| \leq g x$

shows $(\lambda x. \|f x\|) \in O[F](g)$

by (*intro eventually_le_imp_bigo' eventuallyI assms*)

lemma *exp_bigo*:

fixes $f g :: \text{real} \Rightarrow \text{real}$

assumes $\forall_F x \text{ in } \text{at_top}. f x \leq g x$

shows $(\lambda x. \text{exp } (f x)) \in O(\lambda x. \text{exp } (g x))$

proof –

from *assms* **have** $\forall_F x \text{ in } \text{at_top}. \text{exp } (f x) \leq \text{exp } (g x)$ **by** *simp*

hence $\forall_F x \text{ in } \text{at_top}. \|\text{exp } (f x)\| \leq 1 * \|\text{exp } (g x)\|$ **by** *simp*

thus *?thesis* **by** *blast*

qed

lemma *ev_le_imp_exp_bigo*:

fixes $f g :: \text{real} \Rightarrow \text{real}$

assumes *hf*: $\forall_F x \text{ in } \text{at_top}. 0 < f x$

and *hg*: $\forall_F x \text{ in } \text{at_top}. 0 < g x$

and *le*: $\forall_F x \text{ in } \text{at_top}. \ln (f x) \leq \ln (g x)$

shows $f \in O(g)$

proof –

have $\forall_F x \text{ in } \text{at_top}. \text{exp } (\ln (f x)) \leq \text{exp } (\ln (g x))$

using *le* **by** *simp*

hence $\forall_F x \text{ in } \text{at_top}. \|f x\| \leq 1 * \|g x\|$

using *hf hg* **by** *eventually_elim auto*

thus *?thesis* **by** (*intro bigoI*)

qed

lemma *smallo_ln_diverge_1*:

fixes $f :: \text{real} \Rightarrow \text{real}$

assumes *f_ln*: $f \in o(\ln)$

shows $LIM x \text{ at_top}. x * \text{exp } (- f x) :> \text{at_top}$

proof –

have $(\lambda x. \ln x - f x) \sim[\text{at_top}] (\lambda x. \ln x)$

using *assms* **by** (*simp add: asymp_equiv_altdef*)

moreover **have** *filterlim* $(\lambda x. \ln x :: \text{real}) \text{ at_top at_top}$

by *real_asymp*

ultimately **have** *filterlim* $(\lambda x. \ln x - f x) \text{ at_top at_top}$

using *asymp_equiv_at_top_transfer asymp_equiv_sym* **by** *blast*

hence *filterlim* ($\lambda x. \exp (\ln x - f x)$) *at_top at_top*
by (*rule filterlim_compose*[*OF exp_at_top*])
moreover have $\forall_F x$ *in at_top*. $\exp (\ln x - f x) = x * \exp (- f x)$
using *eventually_gt_at_top*[*of 0*]
by *eventually_elim* (*auto simp: exp_diff exp_minus field_simps*)
ultimately show *?thesis*
using *filterlim_cong* **by** *fast*
qed

lemma *ln_ln_asymp_pos*: $\forall_F x :: \text{real}$ *in at_top*. $0 < \ln (\ln x)$ **by** *real_asymp*
lemma *ln_asymp_pos*: $\forall_F x :: \text{real}$ *in at_top*. $0 < \ln x$ **by** *real_asymp*
lemma *x_asymp_pos*: $\forall_F x :: \text{real}$ *in at_top*. $0 < x$ **by** *auto*

1.5 Lemmas of *floor*, *ceil* and *nat_powr*

lemma *nat_le_self*: $0 \leq x \implies \text{nat} (\text{int } x) \leq x$ **by** *auto*
lemma *floor_le*: $\bigwedge x :: \text{real}$. $\lfloor x \rfloor \leq x$ **by** *auto*
lemma *ceil_ge*: $\bigwedge x :: \text{real}$. $x \leq \lceil x \rceil$ **by** *auto*

lemma *nat_lt_real_iff*:
 $(n :: \text{nat}) < (a :: \text{real}) = (n < \text{nat } \lceil a \rceil)$
proof –
have $n < a = (\text{of_int } n < a)$ **by** *auto*
also have $\dots = (n < \lceil a \rceil)$ **by** (*rule less_ceiling_iff* [*symmetric*])
also have $\dots = (n < \text{nat } \lceil a \rceil)$ **by** *auto*
finally show *?thesis* .
qed

lemma *nat_le_real_iff*:
 $(n :: \text{nat}) \leq (a :: \text{real}) = (n < \text{nat } (\lfloor a \rfloor + 1))$
proof –
have $n \leq a = (\text{of_int } n \leq a)$ **by** *auto*
also have $\dots = (n \leq \lfloor a \rfloor)$ **by** (*rule le_floor_iff* [*symmetric*])
also have $\dots = (n < \lfloor a \rfloor + 1)$ **by** *auto*
also have $\dots = (n < \text{nat } (\lfloor a \rfloor + 1))$ **by** *auto*
finally show *?thesis* .
qed

lemma *of_real_nat_power*: $n \text{ nat_powr } (\text{of_real } x :: \text{complex}) = \text{of_real } (n \text{ nat_powr } x)$ **for** $n \ x$
by (*subst of_real_of_nat_eq* [*symmetric*])
 $(\text{subst powr_of_real}, \text{auto})$

lemma *norm_nat_power*: $\|n \text{ nat_powr } (s :: \text{complex})\| = n \text{ powr } (\text{Re } s)$
unfolding *powr_def* **by** *auto*

1.6 Elementary estimation of *exp* and *ln*

lemma *ln_when_ge_3*:
 $1 < \ln x$ **if** $3 \leq x$ **for** $x :: \text{real}$
proof (*rule ccontr*)
assume $\neg 1 < \ln x$
hence $\exp (\ln x) \leq \exp 1$ **by** *auto*
hence $x \leq \exp 1$ **using** *that* **by** *auto*
thus *False* **using** *e_less_272* *that* **by** *auto*
qed

lemma *exp_lemma_1*:

fixes $x :: \text{real}$

assumes $1 \leq x$

shows $1 + \exp x \leq \exp (2 * x)$

proof –

let $?y = \exp x$

have $\ln 2 \leq x$ **using** *assms ln_2_less_1* **by** *auto*

hence $\exp (\ln 2) \leq ?y$ **by** (*subst exp_le_cancel_iff*)

hence $(3 / 2)^2 \leq (?y - 1 / 2)^2$ **by** *auto*

hence $0 \leq -5 / 4 + (?y - 1 / 2)^2$ **by** (*simp add: power2_eq_square*)

also have $\dots = ?y^2 - ?y - 1$ **by** (*simp add: power2_eq_square field_simps*)

finally show *?thesis* **by** (*simp add: exp_double*)

qed

lemma *ln_bound_1*:

fixes $t :: \text{real}$

assumes *Ht*: $0 \leq t$

shows $\ln (14 + 4 * t) \leq 4 * \ln (t + 2)$

proof –

have $\ln (14 + 4 * t) \leq \ln (14 / 2 * (t + 2))$ **using** *Ht* **by** *auto*

also have $\dots = \ln 7 + \ln (t + 2)$ **using** *Ht* **by** (*subst ln_mult*) *auto*

also have $\dots \leq 3 * \ln (t + 2) + \ln (t + 2)$ **proof** –

have $(14 :: \text{real}) \leq 2 \text{ powr } 4$ **by** *auto*

hence $\exp (\ln (14 :: \text{real})) \leq \exp (4 * \ln 2)$

unfolding *powr_def* **by** (*subst exp_ln*) *auto*

hence $\ln (14 :: \text{real}) \leq 4 * \ln 2$ **by** (*subst (asm) exp_le_cancel_iff*)

hence $\ln (14 / 2 :: \text{real}) \leq 3 * \ln 2$ **by** (*subst ln_div*) *auto*

also have $\dots \leq 3 * \ln (t + 2)$ **using** *Ht* **by** *auto*

finally show *?thesis* **by** *auto*

qed

also have $\dots = 4 * \ln (t + 2)$ **by** *auto*

finally show *?thesis* **by** (*auto simp add: field_simps*)

qed

1.7 Miscellaneous lemmas

abbreviation *fds_zeta_complex* :: *complex fds* \equiv *fds_zeta*

lemma *powr_mono_lt_1_cancel*:

fixes $x a b :: \text{real}$

assumes *Hx*: $0 < x \wedge x < 1$

shows $(x \text{ powr } a \leq x \text{ powr } b) = (b \leq a)$

proof –

have $(x \text{ powr } a \leq x \text{ powr } b) = ((x \text{ powr } -1) \text{ powr } -a \leq (x \text{ powr } -1) \text{ powr } -b)$ **by** (*simp add: powr_powr*)

also have $\dots = (-a \leq -b)$ **using** *Hx* **by** (*intro powr_le_cancel_iff*) (*auto simp add: powr_neg_one*)

also have $\dots = (b \leq a)$ **by** *auto*

finally show *?thesis* .

qed

abbreviation *mangoldt_real* :: $_ \Rightarrow \text{real}$ \equiv *mangoldt*

abbreviation *mangoldt_complex* :: $_ \Rightarrow \text{complex}$ \equiv *mangoldt*

lemma *norm_fds_mangoldt_complex*:

$\bigwedge n. \| \text{fds_nth } (\text{fds mangoldt_complex}) n \| = \text{mangoldt_real } n$ **by** (*simp add: fds_nth_fds*)

lemma *suminf_norm_bound*:

```

fixes f :: nat ⇒ 'a :: banach
assumes summable g
  and  $\bigwedge n. \|f n\| \leq g n$ 
shows  $\|suminf f\| \leq (\sum n. g n)$ 
proof -
  have *: summable ( $\lambda n. \|f n\|$ )
    by (rule summable_comparison_test' [where g = g])
      (use assms in auto)
  hence  $\|suminf f\| \leq (\sum n. \|f n\|)$  by (rule summable_norm)
  also have  $(\sum n. \|f n\|) \leq (\sum n. g n)$ 
    by (rule suminf_le) (use assms * in auto)
  finally show ?thesis .
qed

```

```

lemma C1_gt_zero:  $0 < C_1$  unfolding PNT_const_C1_def by auto

```

```

unbundle no_pnt_notation
end
theory Relation_of_PNTs
imports
  PNT_Remainder_Library
begin
unbundle pnt_notation
unbundle prime_counting_notation

```

2 Implication relation of many forms of prime number theorem

```

definition rem_est :: real ⇒ real ⇒ real ⇒ _ where
rem_est c m n ≡  $O(\lambda x. x * exp (-c * ln x powr m * ln (ln x) powr n))$ 

```

```

definition Li :: real ⇒ real where Li x ≡ integral {2..x} ( $\lambda x. 1 / ln x$ )

```

```

definition PNT_1 where PNT_1 c m n ≡  $((\lambda x. \pi x - Li x) \in rem\_est c m n)$ 

```

```

definition PNT_2 where PNT_2 c m n ≡  $((\lambda x. \vartheta x - x) \in rem\_est c m n)$ 

```

```

definition PNT_3 where PNT_3 c m n ≡  $((\lambda x. \psi x - x) \in rem\_est c m n)$ 

```

```

lemma rem_est_compare_powr:
  fixes c m n :: real
  assumes h:  $0 < m m < 1$ 
  shows  $(\lambda x. x powr (2 / 3)) \in rem\_est c m n$ 
  unfolding rem_est_def using assms
  by (cases c 0 :: real rule: linorder_cases; real_asymp)

```

```

lemma PNT_3_imp_PNT_2:
  fixes c m n :: real
  assumes h:  $0 < m m < 1$  and PNT_3 c m n
  shows PNT_2 c m n

```

```

proof -
  have 1:  $(\lambda x. \psi x - x) \in rem\_est c m n$ 
    using assms(3) unfolding PNT_3_def by auto
  have  $(\lambda x. \psi x - \vartheta x) \in O(\lambda x. ln x * sqrt x)$  by (rule  $\psi\_minus\_vartheta\_bigo$ )
  moreover have  $(\lambda x. ln x * sqrt x) \in O(\lambda x. x powr (2 / 3))$  by real_asymp
  ultimately have 2:  $(\lambda x. \psi x - \vartheta x) \in rem\_est c m n$ 
    using rem_est_compare_powr [OF h, of c n] unfolding rem_est_def

```

by (blast intro: landau_o.big.trans)
 have $(\lambda x. \psi x - x - (\psi x - \vartheta x)) \in \text{rem_est_c_m_n}$
 using 1 2 unfolding rem_est_def by (rule sum_in_bigo)
 thus ?thesis unfolding PNT_2_def by simp
 qed

definition r_1 where $r_1 x \equiv \pi x - Li x$ for x

definition r_2 where $r_2 x \equiv \vartheta x - x$ for x

lemma pi_represent_by_theta:

fixes $x :: \text{real}$
 assumes $2 \leq x$
 shows $\pi x = \vartheta x / (\ln x) + \text{integral } \{2..x\} (\lambda t. \vartheta t / (t * (\ln t)^2))$
 proof -
 note integral_unique [OF $\pi_conv_vartheta_integral$]
 with assms show ?thesis by auto
 qed

lemma Li_integrate_by_part:

fixes $x :: \text{real}$
 assumes $2 \leq x$
 shows
 $(\lambda x. 1 / (\ln x)^2) \text{ integrable_on } \{2..x\}$
 $Li x = x / (\ln x) - 2 / (\ln 2) + \text{integral } \{2..x\} (\lambda t. 1 / (\ln t)^2)$
 proof -
 have $(\lambda x. x * (-1 / (x * (\ln x)^2))) \text{ integrable_on } \{2..x\}$
 by (rule integrable_continuous_interval)
 ((rule continuous_intros)+, auto)
 hence $(\lambda x. - (if x = 0 then 0 else 1 / (\ln x)^2)) \text{ integrable_on } \{2..x\}$
 by simp
 moreover have $((\lambda t. 1 / \ln t) \text{ has_vector_derivative } -1 / (t * (\ln t)^2)) \text{ (at } t)$
 when $Ht: 2 \leq t$ for t
 proof -
 define a where $a \equiv (0 * \ln t - 1 * (1 / t)) / (\ln t * \ln t)$
 have $DERIV (\lambda t. 1 / (\ln t)) t :> a$
 unfolding a_def
 proof (rule derivative_intros DERIV_ln_divide)+
 from Ht show $0 < t$ by linarith
 note ln_gt_zero and Ht thus $\ln t \neq 0$ by auto
 qed
 also have $a = -1 / (t * (\ln t)^2)$
 unfolding a_def by (simp add: power2_eq_square)
 finally have $DERIV (\lambda t. 1 / (\ln t)) t :> -1 / (t * (\ln t)^2)$ by auto
 thus ?thesis
 by (subst has_real_derivative_iff_has_vector_derivative [symmetric])
 qed
 ultimately have $((\lambda x. 1 * (1 / \ln x)) \text{ has_integral } x * (1 / \ln x) - 2 * (1 / \ln 2) - \text{integral } \{2..x\} (\lambda x. x * (-1 / (x * (\ln x)^2)))) \{2..x\}$
 using $\langle 2 \leq x \rangle$ by (intro integration_by_part') auto
 note $\beta = \text{this}$ [simplified]
 have $((\lambda x. 1 / \ln x) \text{ has_integral } (x / \ln x - 2 / \ln 2 + \text{integral } \{2..x\} (\lambda x. 1 / (\ln x)^2))) \{2..x\}$
 proof -
 define a where $a t \equiv if t = 0 then 0 else 1 / (\ln t)^2$ for $t :: \text{real}$
 have $\bigwedge t :: \text{real}. t \in \{2..x\} \implies a t = 1 / (\ln t)^2$

unfolding a_def **by** *auto*
hence 4 : $\text{integral } \{2..x\} a = \text{integral } \{2..x\} (\lambda x. 1 / (\ln x)^2)$ **by** (*rule integral_cong*)
from 3 **show** *?thesis*
by (*subst (asm) 4 [unfolded a_def]*)
qed
thus $Li\ x = x / \ln x - 2 / \ln 2 + \text{integral } \{2..x\} (\lambda t. 1 / (\ln t)^2)$ **unfolding** Li_def **by** *auto*
show $(\lambda x. 1 / (\ln x)^2)$ *integrable_on* $\{2..x\}$
by (*rule integrable_continuous_interval*)
((rule continuous_intros)+, auto)
qed

lemma $\vartheta_integrable$:
fixes $x :: real$
assumes $2 \leq x$
shows $(\lambda t. \vartheta\ t / (t * (\ln t)^2))$ *integrable_on* $\{2..x\}$
by (*rule pi_conv_var_integrable [THEN has_integral_integrable], rule assms*)

lemma $r_1_represent_by_r_2$:
fixes $x :: real$
assumes $Hx: 2 \leq x$
shows $(\lambda t. r_2\ t / (t * (\ln t)^2))$ *integrable_on* $\{2..x\}$ (**is** $?P$)
 $r_1\ x = r_2\ x / (\ln x) + 2 / \ln 2 + \text{integral } \{2..x\} (\lambda t. r_2\ t / (t * (\ln t)^2))$ (**is** $?Q$)
proof –
have 0 : $\bigwedge t. t \in \{2..x\} \implies (\vartheta\ t - t) / (t * (\ln t)^2) = \vartheta\ t / (t * (\ln t)^2) - 1 / (\ln t)^2$
by (*subst diff_divide_distrib, auto*)
note $integrables = \vartheta_integrable\ Li_integrate_by_part(1)$
let $?D = \text{integral } \{2..x\} (\lambda t. \vartheta\ t / (t * (\ln t)^2)) -$
 $\text{integral } \{2..x\} (\lambda t. 1 / (\ln t)^2)$
have $((\lambda t. \vartheta\ t / (t * (\ln t)^2) - 1 / (\ln t)^2)$ *has_integral*
 $?D)$ $\{2..x\}$
unfolding r_2_def **by**
(rule has_integral_diff)
(rule integrables [THEN integrable_integral], rule Hx)+
hence 0 : $((\lambda t. r_2\ t / (t * (\ln t)^2))$ *has_integral*
 $?D)$ $\{2..x\}$
unfolding r_2_def **by** (*subst has_integral_cong [OF 0]*)
thus $?P$ **by** (*rule has_integral_integrable*)
note $1 = 0$ [*THEN integral_unique*]
have 2 : $r_2\ x / \ln x = \vartheta\ x / \ln x - x / \ln x$
unfolding r_2_def **by** (*rule diff_divide_distrib*)
from $pi_represent_by_theta$ **and** $Li_integrate_by_part(2)$ **and** *assms*
have $\pi\ x - Li\ x = \vartheta\ x / \ln x$
 $+ \text{integral } \{2..x\} (\lambda t. \vartheta\ t / (t * (\ln t)^2))$
 $- (x / \ln x - 2 / \ln 2 + \text{integral } \{2..x\} (\lambda t. 1 / (\ln t)^2))$
by *auto*
also have $\dots = r_2\ x / \ln x + 2 / \ln 2$
 $+ \text{integral } \{2..x\} (\lambda t. r_2\ t / (t * (\ln t)^2))$
by (*subst 2, subst 1*) *auto*
finally show $?Q$ **unfolding** r_1_def **by** *auto*
qed

lemma $exp_integral_asympt$:
fixes $f\ f' :: real \Rightarrow real$
assumes cf : *continuous_on* $\{a.. \}$ f
and der : $\bigwedge x. a < x \implies DERIV\ f\ x := f'\ x$

and $td: ((\lambda x. x * f' x) \longrightarrow 0) \text{ at_top}$
and $f_ln: f \in o(ln)$
shows $(\lambda x. \text{integral } \{a..x\} (\lambda t. \text{exp } (-f t))) \sim[at_top] (\lambda x. x * \text{exp}(-f x))$
proof (*rule asymp_equivI'*, *rule lhospital_at_top_at_top*)
have $cont_exp: \text{continuous_on } \{a..\} (\lambda t. \text{exp } (-f t))$
using *cf by (intro continuous_intros)*
show $\forall_F x \text{ in } at_top. ((\lambda x. \text{integral } \{a..x\} (\lambda t. \text{exp } (-f t)))$
 $\text{has_real_derivative } \text{exp } (-f x)) (\text{at } x) \text{ (is eventually ?P ?F)}$
proof (*rule eventually_at_top_linorderI'*)
fix x **assume** $1: a < x$
hence $2: a \leq x$ **by** *linarith*
have $3: (\text{at } x \text{ within } \{a..x+1\}) = (\text{at } x)$
by (*rule at_within_interior*) (*auto intro: 1*)
show $?P x$
by (*subst 3 [symmetric]*, *rule integral_has_real_derivative*)
(rule continuous_on_subset [OF cont_exp], auto intro: 2)
qed
have $\forall_F x \text{ in } at_top. ((\lambda x. x * \text{exp } (-f x))$
 $\text{has_real_derivative } 1 * \text{exp } (-f x) + \text{exp } (-f x) * (-f' x) * x) (\text{at } x)$
(is eventually ?P ?F)
proof (*rule eventually_at_top_linorderI'*)
fix x **assume** $1: a < x$
hence $2: (\text{at } x \text{ within } \{a<..\}) = (\text{at } x)$ **by** (*auto intro: at_within_open*)
show $?P x$
by (*subst 2 [symmetric]*, *intro derivative_intros*)
(subst 2, rule der, rule 1)
qed
moreover have
 $1 * \text{exp } (-f x) + \text{exp } (-f x) * (-f' x) * x$
 $= \text{exp } (-f x) * (1 - x * f' x)$ **for** $x :: \text{real}$
by (*simp add: field_simps*)
ultimately show $\forall_F x \text{ in } at_top.$
 $((\lambda x. x * \text{exp } (-f x))$
 $\text{has_real_derivative } \text{exp } (-f x) * (1 - x * f' x)) (\text{at } x)$ **by** *auto*
show $LIM x \text{ at_top. } x * \text{exp } (-f x) :> at_top$
using f_ln **by** (*rule smallo_ln_diverge_1*)
have $((\lambda x. 1 / (1 - x * f' x)) \longrightarrow 1 / (1 - 0)) \text{ at_top}$
by (*(rule tendsto_intros)+, rule td, linarith*)
thus $((\lambda x. \text{exp } (-f x) / (\text{exp } (-f x) * (1 - x * f' x))) \longrightarrow 1) \text{ at_top}$ **by** *auto*
have $((\lambda x. 1 - x * f' x) \longrightarrow 1 - 0) \text{ at_top}$
by (*(rule tendsto_intros)+, rule td*)
hence $0: ((\lambda x. 1 - x * f' x) \longrightarrow 1) \text{ at_top}$ **by** *simp*
hence $\forall_F x \text{ in } at_top. 0 < 1 - x * f' x$
by (*rule order_tendstoD*) *linarith*
moreover have $\forall_F x \text{ in } at_top. 0 < 1 - x * f' x \longrightarrow \text{exp } (-f x) * (1 - x * f' x) \neq 0$ **by** *auto*
ultimately show $\forall_F x \text{ in } at_top. \text{exp } (-f x) * (1 - x * f' x) \neq 0$
by (*rule eventually_rev_mp*)
qed

lemma *x_mul_exp_larger_than_const:*

fixes $c :: \text{real}$ **and** $g :: \text{real} \Rightarrow \text{real}$

assumes $g_ln: g \in o(ln)$

shows $(\lambda x. c) \in O(\lambda x. x * \text{exp}(-g x))$

proof –

have $LIM x \text{ at_top. } x * \text{exp } (-g x) :> at_top$

using g_ln by (rule *smallo_ln_diverge_1*)
 hence $\forall_F x$ in *at_top*. $1 \leq x * \exp(-g x)$
 using *filterlim_at_top* by *fast*
 hence $\forall_F x$ in *at_top*. $\|c\| * 1 \leq \|c\| * \|x * \exp(-g x)\|$
 by (rule *eventually_rev_mp*)
 (auto simp del: *mult_1_right*
 intro!: eventuallyI mult_left_mono)
 thus $(\lambda x. c :: real) \in O(\lambda x. x * \exp(-g x))$ by *auto*
 qed

lemma *integral_bigo_exp'*:

fixes $a :: real$ and $f g g' :: real \Rightarrow real$
 assumes *f_bound*: $f \in O(\lambda x. \exp(-g x))$
 and *Hf*: $\bigwedge x. a \leq x \implies f$ *integrable_on* $\{a..x\}$
 and *Hf'*: $\bigwedge x. a \leq x \implies (\lambda x. |f x|)$ *integrable_on* $\{a..x\}$
 and *Hg*: *continuous_on* $\{a..\}$ g
 and *der*: $\bigwedge x. a < x \implies DERIV g x :> g' x$
 and *td*: $((\lambda x. x * g' x) \longrightarrow 0)$ *at_top*
 and *g_ln*: $g \in o(ln)$
 shows $(\lambda x. \text{integral}\{a..x\} f) \in O(\lambda x. x * \exp(-g x))$
 proof –
 have $\bigwedge y. \text{continuous_on } \{a..y\} g$
 by (rule *continuous_on_subset*, rule *Hg*) *auto*
 hence $\bigwedge y. (\lambda x. \exp(-g x))$ *integrable_on* $\{a..y\}$
 by (*intro integrable_continuous_interval*)
 (rule *continuous_intros*)+
 hence $\bigwedge y. (\lambda x. |\exp(-g x)|)$ *integrable_on* $\{a..y\}$ by *simp*
 hence $(\lambda x. \text{integral}\{a..x\} f) \in O(\lambda x. 1 + \text{integral}\{a..x\} (\lambda x. |\exp(-g x)|))$
 using *assms* by (*intro integral_bigo*)
 hence $(\lambda x. \text{integral}\{a..x\} f) \in O(\lambda x. 1 + \text{integral}\{a..x\} (\lambda x. \exp(-g x)))$ by *simp*
 also have $(\lambda x. 1 + \text{integral}\{a..x\} (\lambda x. \exp(-g x))) \in O(\lambda x. x * \exp(-g x))$
 proof (rule *sum_in_bigo*)
 show $(\lambda x. 1 :: real) \in O(\lambda x. x * \exp(-g x))$
 by (*intro x_mul_exp_larger_than_const g_ln*)
 show $(\lambda x. \text{integral } \{a..x\} (\lambda x. \exp(-g x))) \in O(\lambda x. x * \exp(-g x))$
 by (rule *asympt_equiv_imp_bigo*, rule *exp_integral_asympt*, auto *intro: assms*)
 qed
 finally show *?thesis* .
 qed

lemma *integral_bigo_exp*:

fixes $a b :: real$ and $f g g' :: real \Rightarrow real$
 assumes *le*: $a \leq b$
 and *f_bound*: $f \in O(\lambda x. \exp(-g x))$
 and *Hf*: $\bigwedge x. a \leq x \implies f$ *integrable_on* $\{a..x\}$
 and *Hf'*: $\bigwedge x. b \leq x \implies (\lambda x. |f x|)$ *integrable_on* $\{b..x\}$
 and *Hg*: *continuous_on* $\{b..\}$ g
 and *der*: $\bigwedge x. b < x \implies DERIV g x :> g' x$
 and *td*: $((\lambda x. x * g' x) \longrightarrow 0)$ *at_top*
 and *g_ln*: $g \in o(ln)$
 shows $(\lambda x. \text{integral } \{a..x\} f) \in O(\lambda x. x * \exp(-g x))$
 proof –
 have $(\lambda x. \text{integral } \{a..b\} f) \in O(\lambda x. x * \exp(-g x))$
 by (*intro x_mul_exp_larger_than_const g_ln*)
 moreover have $(\lambda x. \text{integral } \{b..x\} f) \in O(\lambda x. x * \exp(-g x))$

by (*intro integral_bigo_exp'* [**where** $?g' = g$]
f_bound Hf Hf' Hg der td g_ln)
(use le Hf integrable_cut' in auto)
ultimately have $(\lambda x. \text{integral } \{a..b\} f + \text{integral } \{b..x\} f) \in O(\lambda x. x * \exp(-g x))$
by (*rule sum_in_bigo*)
moreover have $\text{integral } \{a..x\} f = \text{integral } \{a..b\} f + \text{integral } \{b..x\} f$ **when** $b \leq x$ **for** x
by (*subst eq_commute, rule Henstock_Kurzweil_Integration.integral_combine*)
(insert le that, auto intro: Hf)
hence $\forall_F x \text{ in } \text{at_top}. \text{integral } \{a..x\} f = \text{integral } \{a..b\} f + \text{integral } \{b..x\} f$
by (*rule eventually_at_top_linorderI*)
ultimately show *?thesis*
by (*simp add: landau_o.big.in_cong*)
qed

lemma *integrate_r2_estimate:*

fixes $c m n :: \text{real}$
assumes $hm: 0 < m \ m < 1$
and $h: r_2 \in \text{rem_est } c m n$
shows $(\lambda x. \text{integral } \{2..x\} (\lambda t. r_2 t / (t * (\ln t)^2))) \in \text{rem_est } c m n$
unfolding *rem_est_def*
proof (*subst mult.assoc,*
subst minus_mult_left [symmetric],
rule integral_bigo_exp)
show $(2 :: \text{real}) \leq 3$ **by** *auto*
show $(\lambda x. c * (\ln x \text{ powr } m * \ln (\ln x) \text{ powr } n)) \in o(\ln)$
using hm **by** *real_asymp*
have $\ln x \neq 1$ **when** $3 \leq x$ **for** $x :: \text{real}$
using *ln_when_ge_3 [of x] that* **by** *auto*
thus *continuous_on* $\{3..\}$ $(\lambda x. c * (\ln x \text{ powr } m * \ln (\ln x) \text{ powr } n))$
by (*intro continuous_intros*) *auto*
show $(\lambda t. r_2 t / (t * (\ln t)^2)) \text{ integrable_on } \{2..x\}$
if $2 \leq x$ **for** x **using** *that* **by** (*rule r1_represent_by_r2(1)*)
define g **where** $g x \equiv$
 $c * (m * \ln x \text{ powr } (m - 1) * (1 / x * 1) * \ln (\ln x) \text{ powr } n$
 $+ n * \ln (\ln x) \text{ powr } (n - 1) * (1 / \ln x * (1 / x)) * \ln x \text{ powr } m)$
for x
show $((\lambda x. c * (\ln x \text{ powr } m * \ln (\ln x) \text{ powr } n)) \text{ has_real_derivative } g x)$ *(at x)*
if $3 < x$ **for** x
proof –
have $*$: *at x within* $\{3<..\}$ $= \text{at } x$
by (*rule at_within_open*) *(auto intro: that)*
moreover have
 $((\lambda x. c * (\ln x \text{ powr } m * \ln (\ln x) \text{ powr } n)) \text{ has_real_derivative } g x)$
(at x within $\{3<..\}$ *)*
unfolding g_def **using** *that*
by (*intro derivative_intros DERIV_mult DERIV_cmult*)
*(auto intro: ln_when_ge_3 DERIV_ln_divide simp add: *)*
ultimately show *?thesis* **by** *auto*
qed
show $((\lambda x. x * g x) \longrightarrow 0)$ *at_top*
unfolding g_def **using** hm **by** *real_asymp*
have $nz: \forall_F t :: \text{real in } \text{at_top}. t * (\ln t)^2 \neq 0$
proof (*rule eventually_at_top_linorderI'*)
fix $x :: \text{real}$ **assume** $1 < x$
thus $x * (\ln x)^2 \neq 0$ **by** *auto*

qed

define h **where** $h\ x \equiv \exp(-c * \ln\ x\ \text{powr}\ m * \ln(\ln\ x)\ \text{powr}\ n)$ **for** x

have $(\lambda t. r_2\ t / (t * (\ln\ t)^2)) \in O(\lambda x. (x * h\ x) / (x * (\ln\ x)^2))$

by $(\text{rule}\ \text{landau_o.big.divide_right}, \text{rule}\ \text{nz})$
 $(\text{unfold}\ h_def, \text{fold}\ \text{rem_est_def}, \text{rule}\ h)$

also have $(\lambda x. (x * h\ x) / (x * (\ln\ x)^2)) \in O(\lambda x. h\ x)$

proof –

have $(\lambda x :: \text{real}. 1 / (\ln\ x)^2) \in O(\lambda x. 1)$ **by** real_asympt

hence $(\lambda x. h\ x * (1 / (\ln\ x)^2)) \in O(\lambda x. h\ x * 1)$

by $(\text{rule}\ \text{landau_o.big.mult_left})$

thus $?thesis$

by $(\text{auto}\ \text{simp}\ \text{add}: \text{field_simps})$
 $\text{intro!}: \text{landau_o.big.ev_eq_trans2})$
 $(\text{auto}\ \text{simp}: \text{frequently_def}\ \text{intro}: \text{eventually_at_top_linorderI}\ [\text{of}\ 1])$

qed

finally show $(\lambda t. r_2\ t / (t * (\ln\ t)^2))$
 $\in O(\lambda x. \exp(-c * (\ln\ x\ \text{powr}\ m * \ln(\ln\ x)\ \text{powr}\ n)))$

unfolding h_def **by** $(\text{simp}\ \text{add}: \text{algebra_simps})$

have $(\lambda x. r_2\ x / (x * (\ln\ x)^2))$ **absolutely_integrable_on** $\{2..x\}$

if $*: 2 \leq x$ **for** x

proof $(\text{rule}\ \text{set_integrableI_bounded})$

show $\{2..x\} \in \text{sets}\ \text{lebesgue}$ **by** auto

show $\text{emeasure}\ \text{lebesgue}\ \{2..x\} < \infty$ **using** $*$ **by** auto

have $(\lambda t. r_2\ t / (t * (\ln\ t)^2) * \text{indicator}\ \{2..x\}\ t) \in \text{borel_measurable}\ \text{lebesgue}$

using $*$ **by** $(\text{intro}\ \text{integrable_integral})$
 $[\text{THEN}\ \text{has_integral_implies_lebesgue_measurable_real}]$
 $(\text{rule}\ r_1_represent_by_r_2(1))$

thus $(\lambda t. \text{indicat_real}\ \{2..x\}\ t *_{\mathbb{R}} (r_2\ t / (t * (\ln\ t)^2))) \in \text{borel_measurable}\ \text{lebesgue}$

by $(\text{simp}\ \text{add}: \text{mult_ac})$

let $?C = (\ln\ 4 + 1) / (\ln\ 2)^2 :: \text{real}$

show $AE\ t \in \{2..x\}$ **in** $\text{lebesgue}. \|r_2\ t / (t * (\ln\ t)^2)\| \leq ?C$

proof $(\text{rule}\ AE_I2, \text{safe})$

fix t **assume** $t \in \{2..x\}$

hence $h: 1 \leq t / 2 \leq t$ **by** auto

hence $0 \leq \vartheta\ t \wedge \vartheta\ t < \ln\ 4 * t$ **by** $(\text{auto}\ \text{intro}: \vartheta_upper_bound)$

hence $*: |\vartheta\ t| \leq \ln\ 4 * t$ **by** auto

have $1 \leq \ln\ t / \ln\ 2$ **using** h **by** auto

hence $1 \leq (\ln\ t / \ln\ 2)^2$ **by** auto

also have $\dots = (\ln\ t)^2 / (\ln\ 2)^2$ **unfolding** power2_eq_square **by** auto

finally have $1 \leq (\ln\ t)^2 / (\ln\ 2)^2$.

hence $|r_2\ t| \leq |\vartheta\ t| + |t|$ **unfolding** r_2_def **by** auto

also have $\dots \leq \ln\ 4 * t + 1 * t$ **using** $h * \text{by}\ \text{auto}$

also have $\dots = (\ln\ 4 + 1) * t$ **by** $(\text{simp}\ \text{add}: \text{algebra_simps})$

also have $\dots \leq (\ln\ 4 + 1) * t * ((\ln\ t)^2 / (\ln\ 2)^2)$

by $(\text{auto}\ \text{simp}\ \text{add}: \text{field_simps})$
 $(\text{rule}\ \text{add_mono}; \text{rule}\ \text{rev_mp}[\text{OF}\ h(2)], \text{auto})$

finally have $*: |r_2\ t| \leq ?C * (t * (\ln\ t)^2)$ **by** auto

thus $\|r_2\ t / (t * (\ln\ t)^2)\| \leq ?C$

using $h * \text{by}\ (\text{auto}\ \text{simp}\ \text{add}: \text{field_simps})$

qed

qed

hence $\bigwedge x. 2 \leq x \implies (\lambda x. |r_2\ x / (x * (\ln\ x)^2)|)$ **integrable_on** $\{2..x\}$

by $(\text{fold}\ \text{real_norm_def})$
 $(\text{rule}\ \text{absolutely_integrable_on_def}\ [\text{THEN}\ \text{iffD1}, \text{THEN}\ \text{conjunct2}])$

thus $\bigwedge x. 3 \leq x \implies (\lambda x. |r_2\ x / (x * (\ln\ x)^2)|)$ **integrable_on** $\{3..x\}$

using $\langle 2 \leq 3 \rangle$ *integrable_cut'* by blast
qed

lemma *r2_div_ln_estimate*:

fixes $c\ m\ n :: \text{real}$
 assumes $hm: 0 < m\ m < 1$
 and $h: r_2 \in \text{rem_est}\ c\ m\ n$
 shows $(\lambda x. r_2\ x / (\ln\ x) + 2 / \ln\ 2) \in \text{rem_est}\ c\ m\ n$

proof –

have $(\lambda x. r_2\ x / \ln\ x) \in O(r_2)$
 proof (intro *bigoI eventually_at_top_linorderI*)
 fix $x :: \text{real}$ assume $1:\text{exp}\ 1 \leq x$
 have $2:(0 :: \text{real}) < \text{exp}\ 1$ by *simp*
 hence $3:0 < x$ using 1 by *linarith*
 have $4: 0 \leq |r_2\ x|$ by *auto*
 have $(1 :: \text{real}) = \ln(\text{exp}\ 1)$ by *simp*
 also have $\dots \leq \ln\ x$ using $1\ 2\ 3$ by (*subst ln_le_cancel_iff*)
 finally have $1 \leq \ln\ x$.
 thus $\|r_2\ x / \ln\ x\| \leq 1 * \|r_2\ x\|$
 by (*auto simp add: field_simps, subst mult_le_cancel_right1, auto*)

qed

with h have $1: (\lambda x. r_2\ x / \ln\ x) \in \text{rem_est}\ c\ m\ n$
 unfolding *rem_est_def* using *landau_o.big_trans* by blast
 moreover have $(\lambda x :: \text{real}. 2 / \ln\ 2) \in O(\lambda x. x\ \text{powr}\ (2 / 3))$
 by *real_asymp*
 hence $(\lambda x :: \text{real}. 2 / \ln\ 2) \in \text{rem_est}\ c\ m\ n$
 using *rem_est_compare_powr* [OF hm , of $c\ n$]
 unfolding *rem_est_def* by (*rule landau_o.big.trans*)
 ultimately show *?thesis*
 unfolding *rem_est_def* by (*rule sum_in_bigo*)

qed

lemma *PNT_2_imp_PNT_1*:

fixes $l :: \text{real}$
 assumes $h: 0 < m\ m < 1$ and *PNT_2* $c\ m\ n$
 shows *PNT_1* $c\ m\ n$

proof –

from *assms*(3) have $h': r_2 \in \text{rem_est}\ c\ m\ n$
 unfolding *PNT_2_def r2_def* by *auto*
 let $?a = \lambda x. r_2\ x / \ln\ x + 2 / \ln\ 2$
 let $?b = \lambda x. \text{integral}\ \{2..x\}\ (\lambda t. r_2\ t / (t * (\ln\ t)^2))$
 have $1: \forall_F\ x\ \text{in}\ \text{at_top}. \pi\ x - Li\ x = ?a\ x + ?b\ x$
 by (*rule eventually_at_top_linorderI, fold r1_def*)
 (*rule r1_represent_by_r2(2), blast*)
 have $2: (\lambda x. ?a\ x + ?b\ x) \in \text{rem_est}\ c\ m\ n$
 by (*unfold rem_est_def, (rule sum_in_bigo; fold rem_est_def)*)
 (*intro r2_div_ln_estimate integrate_r2_estimate h h'*)
 from *landau_o.big.in_cong* [OF 1] and 2 show *?thesis*
 unfolding *PNT_1_def rem_est_def* by blast

qed

theorem *PNT_3_imp_PNT_1*:

fixes $l :: \text{real}$
 assumes $h: 0 < m\ m < 1$ and *PNT_3* $c\ m\ n$
 shows *PNT_1* $c\ m\ n$

by (intro PNT_2_imp_PNT_1 PNT_3_imp_PNT_2 assms)

```
hide_const (open) r1 r2
unbundle no_prime_counting_notation
unbundle no_pnt_notation
end
theory PNT_Complex_Analysis_Lemmas
imports
  PNT_Remainder_Library
begin
unbundle pnt_notation
```

3 Some basic theorems in complex analysis

3.1 Introduction rules for holomorphic functions and analytic functions

```
lemma holomorphic_on_shift [holomorphic_intros]:
  assumes f holomorphic_on ((λz. s + z) ` A)
  shows (λz. f (s + z)) holomorphic_on A
proof -
  have (f ∘ (λz. s + z)) holomorphic_on A
    using assms by (intro holomorphic_on_compose holomorphic_intros)
  thus ?thesis unfolding comp_def by auto
qed
```

```
lemma holomorphic_logderiv [holomorphic_intros]:
  assumes f holomorphic_on A open A ∧ z. z ∈ A ⇒ f z ≠ 0
  shows (λs. logderiv f s) holomorphic_on A
  using assms unfolding logderiv_def by (intro holomorphic_intros)
```

```
lemma holomorphic_glue_to_analytic:
  assumes o: open S open T
  and hf: f holomorphic_on S
  and hg: g holomorphic_on T
  and hI: ∧z. z ∈ S ⇒ z ∈ T ⇒ f z = g z
  and hU: U ⊆ S ∪ T
  obtains h
  where h analytic_on U
  ∧z. z ∈ S ⇒ h z = f z
  ∧z. z ∈ T ⇒ h z = g z
proof -
  define h where h z ≡ if z ∈ S then f z else g z for z
  show ?thesis proof
    have h holomorphic_on S ∪ T
      unfolding h_def by (rule holomorphic_on_Iif_Un) (use assms in auto)
    thus h analytic_on U
      by (subst analytic_on_holomorphic) (use hU o in auto)
  next
    fix z assume *:z ∈ S
    show h z = f z unfolding h_def using * by auto
  next
    fix z assume *:z ∈ T
    show h z = g z unfolding h_def using * hI by auto
qed
qed
```

lemma *analytic_on_powr_right* [*analytic_intros*]:
 assumes *f analytic_on s*
 shows $(\lambda z. w \text{ powr } f z)$ *analytic_on s*
proof (*cases w = 0*)
 case *False*
 with *assms* **show** *?thesis*
 unfolding *analytic_on_def holomorphic_on_def field_differentiable_def*
 by (*metis (full_types) DERIV_chain' has_field_derivative_powr_right*)
qed *simp*

3.2 Factorization of analytic function on compact region

definition *not_zero_on* (**infixr** *not'_zero'_on* 46)
 where *f not_zero_on S* $\equiv \exists z \in S. f z \neq 0$

lemma *not_zero_on_obtain*:
 assumes *f not_zero_on S* and $S \subseteq T$
 obtains *t* where $f t \neq 0$ and $t \in T$
using *assms* **unfolding** *not_zero_on_def* **by** *auto*

lemma *analytic_on_holomorphic_connected*:
 assumes *hf: f analytic_on S*
 and *con: connected A*
 and *ne: $\xi \in A$ and AS: $A \subseteq S$*
 obtains $T T'$ where
f holomorphic_on T f holomorphic_on T'
open T open T' A \subseteq T S \subseteq T' connected T
proof –
 obtain T'
 where *oT': open T' and sT': $S \subseteq T'$*
 and *holf': f holomorphic_on T'*
 using *analytic_on_holomorphic hf* **by** *blast*
define T where $T \equiv \text{connected_component_set } T' \xi$
have $TT': T \subseteq T'$ **unfolding** *T_def* **by** (*rule connected_component_subset*)
hence *holf: f holomorphic_on T* **using** *holf'* **by** *auto*
have *opT: open T* **unfolding** *T_def* **using** *oT'* **by** (*rule open_connected_component*)
have *conT: connected T* **unfolding** *T_def* **by** (*rule connected_connected_component*)
have $A \subseteq T'$ **using** *AS sT'* **by** *blast*
hence $AT: A \subseteq T$ **unfolding** *T_def* **using** *ne con* **by** (*intro connected_component_maximal*)
show *?thesis* **using** *holf holf' opT oT' AT sT' conT* **that** **by** *blast*
qed

lemma *analytic_factor_zero*:
 assumes *hf: f analytic_on S*
 and $KS: K \subseteq S$ and *con: connected K*
 and $\xi K: \xi \in K$ and $\xi z: f \xi = 0$
 and *nz: f not_zero_on K*
 obtains $g r n$
 where $0 < n$ $0 < r$
g analytic_on S g not_zero_on K
 $\bigwedge z. z \in S \implies f z = (z - \xi)^{\wedge n} * g z$
 $\bigwedge z. z \in \text{ball } \xi r \implies g z \neq 0$
proof –
have *f analytic_on S connected K*
 $\xi \in K K \subseteq S$ **using** *assms* **by** *auto*

then obtain $T T'$
where $hol_f: f \text{ holomorphic_on } T$
and $hol_{f'}: f \text{ holomorphic_on } T'$
and $opT: \text{open } T$ **and** $opT': \text{open } T'$
and $KT: K \subseteq T$ **and** $ST': S \subseteq T'$
and $conT: \text{connected } T$
by $(\text{rule analytic_on_holomorphic_connected})$
obtain η **where** $f\eta: f \eta \neq 0$ **and** $\eta K: \eta \in K$
using nz **by** $(\text{rule not_zero_on_obtain, blast})$
hence $\xi T: \xi \in T$ **and** $\xi T': \xi \in T'$
and $\eta T: \eta \in T$ **using** $\xi K \eta K KT KS ST'$ **by** $blast+$
hence $nc: \neg f \text{ constant_on } T$ **using** $f\eta \xi z$ **unfolding** constant_on_def **by** $fastforce$
obtain $g r n$
where $1: 0 < n$ **and** $2: 0 < r$
and $bT: \text{ball } \xi r \subseteq T$
and $hg: g \text{ holomorphic_on ball } \xi r$
and $fw: \bigwedge z. z \in \text{ball } \xi r \implies f z = (z - \xi) ^ n * g z$
and $gw: \bigwedge z. z \in \text{ball } \xi r \implies g z \neq 0$
by $(\text{rule holomorphic_factor_zero_nonconstant, (rule hol_opT conT } \xi T \xi z nc)+, blast)$
have $sT: S \subseteq T' - \{\xi\} \cup \text{ball } \xi r$ **using** $2 ST'$ **by** $auto$
have $hz: (\lambda z. f z / (z - \xi) ^ n) \text{ holomorphic_on } (T' - \{\xi\})$
using $hol_{f'}$ **by** $((\text{intro holomorphic_intros})+, auto)$
obtain h
where $3: h \text{ analytic_on } S$
and $hf: \bigwedge z. z \in T' - \{\xi\} \implies h z = f z / (z - \xi) ^ n$
and $hb: \bigwedge z. z \in \text{ball } \xi r \implies h z = g z$
by $(\text{rule holomorphic_glue_to_analytic}$
[where $f = \lambda z. f z / (z - \xi) ^ n$ **and**
 $g = g$ **and** $S = T' - \{\xi\}$ **and** $T = \text{ball } \xi r$ **and** $U = S]$
 $(\text{use } opT' 2 ST' hg fw hz \text{ in } \langle auto \text{ simp add: holomorphic_intros} \rangle)$
have $\xi \in \text{ball } \xi r$ **using** 2 **by** $auto$
hence $h \xi \neq 0$ **using** $hb gw 2$ **by** $auto$
hence $4: h \text{ not_zero_on } K$ **unfolding** not_zero_on_def **using** ξK **by** $auto$
have $5: f z = (z - \xi) ^ n * h z$ **if** $*$: $z \in S$ **for** z
proof –
consider $z = \xi \mid z \in S - \{\xi\}$ **using** $*$ **by** $auto$
thus $?thesis$ **proof cases**
assume $*$: $z = \xi$
show $?thesis$ **using** $\xi z 1$ **by** $(\text{subst } (1 2) *, auto)$
next
assume $*$: $z \in S - \{\xi\}$
show $?thesis$ **using** $hf ST' *$ **by** $(\text{auto simp add: field_simps})$
qed
qed
have $6: \bigwedge w. w \in \text{ball } \xi r \implies h w \neq 0$ **using** $hb gw$ **by** $auto$
show $?thesis$ **by** $((\text{standard; rule } 1 2 3 4 5 6), blast+)$
qed

lemma $\text{analytic_compact_finite_zeros}$:

assumes $af: f \text{ analytic_on } S$
and $KS: K \subseteq S$
and $con: \text{connected } K$
and $cm: \text{compact } K$
and $nz: f \text{ not_zero_on } K$
shows $\text{finite } \{z \in K. f z = 0\}$

proof (*cases f constant_on K*)
assume *: *f constant_on K*
have $\bigwedge z. z \in K \implies f z \neq 0$ **using** *nz * unfolding not_zero_on_def constant_on_def* **by** *auto*
hence **: $\{z \in K. f z = 0\} = \{\}$ **by** *auto*
thus *?thesis* **by** (*subst **, auto*)
next
assume *: $\neg f \text{ constant_on } K$
obtain ξ **where** *ne: $\xi \in K$* **using** *not_zero_on_obtain nz* **by** *blast*
obtain $T T'$ **where** *opT: open T* **and** *conT: connected T*
and *ST: $K \subseteq T$* **and** *holf: f holomorphic_on T*
and *f holomorphic_on T'*
by (*metis af KS con ne analytic_on_holomorphic_connected*)
have $\neg f \text{ constant_on } T$ **using** *ST * unfolding constant_on_def* **by** *blast*
thus *?thesis* **using** *holf opT conT cm ST* **by** (*intro holomorphic_compact_finite_zeros*)
qed

3.2.1 Auxiliary propositions for theorem *analytic_factorization*

definition *analytic_factor_p'* **where**
 $\langle \text{analytic_factor_p}' f S K \equiv$
 $\exists g n. \exists \alpha :: \text{nat} \Rightarrow \text{complex}.$
 $g \text{ analytic_on } S$
 $\wedge (\forall z \in K. g z \neq 0)$
 $\wedge (\forall z \in S. f z = g z * (\prod_{k < n. z - \alpha k})$
 $\wedge \alpha \text{ ' } \{..<n\} \subseteq K \rangle$

definition *analytic_factor_p* **where**
 $\langle \text{analytic_factor_p } F \equiv$
 $\forall f S K. f \text{ analytic_on } S$
 $\longrightarrow K \subseteq S$
 $\longrightarrow \text{connected } K$
 $\longrightarrow \text{compact } K$
 $\longrightarrow f \text{ not_zero_on } K$
 $\longrightarrow \{z \in K. f z = 0\} = F$
 $\longrightarrow \text{analytic_factor_p}' f S K \rangle$

lemma *analytic_factorization_E*:

shows *analytic_factor_p* $\{\}$

unfolding *analytic_factor_p_def*

proof (*intro conjI allI impI*)

fix $f S K$
assume *af: f analytic_on S*
and *KS: $K \subseteq S$*
and *con: connected K*
and *cm: compact K*
and *nz: $\{z \in K. f z = 0\} = \{\}$*

show *analytic_factor_p'* $f S K$

unfolding *analytic_factor_p'_def*

proof (*intro ballI conjI exI*)

show $f \text{ analytic_on } S \bigwedge z. z \in K \implies f z \neq 0$

$\bigwedge z. z \in S \implies f z = f z * (\prod_{k < (0 :: \text{nat}). z - (\lambda_. 0) k}$

by (*rule af, use nz in auto*)

show $(\lambda k :: \text{nat}. 0) \text{ ' } \{..<0\} \subseteq K$ **by** *auto*

qed

qed

```

lemma analytic_factorization_I:
  assumes ind: analytic_factor_p F
    and  $\xi \notin F$ 
  shows analytic_factor_p (insert  $\xi$  F)
unfolding analytic_factor_p_def
proof (intro allI impI)
  fix f S K
  assume af: f analytic_on S
    and KS:  $K \subseteq S$ 
    and con: connected K
    and nz: f not_zero_on K
    and cm: compact K
    and zr:  $\{z \in K. f z = 0\} = \text{insert } \xi F$ 
  show analytic_factor_p' f S K
proof -
  have f analytic_on S  $K \subseteq S$  connected K
     $\xi \in K$   $f \xi = 0$  f not_zero_on K
  using af KS con zr nz by auto
  then obtain h r k
  where  $0 < k$  and  $0 < r$  and ah: h analytic_on S
    and nh: h not_zero_on K
    and f_z:  $\bigwedge z. z \in S \implies f z = (z - \xi) ^ k * h z$ 
    and ball:  $\bigwedge z. z \in \text{ball } \xi r \implies h z \neq 0$ 
  by (rule analytic_factor_zero) blast
  hence h $\xi$ :  $h \xi \neq 0$  using ball by auto
  hence  $\bigwedge z. z \in K \implies h z = 0 \iff f z = 0 \wedge z \neq \xi$  by (subst f_z) (use KS in auto)
  hence  $\{z \in K. h z = 0\} = \{z \in K. f z = 0\} - \{\xi\}$  by auto
  also have ... = F by (subst zr, intro Diff_insert_absorb  $\xi ni$ )
  finally have  $\{z \in K. h z = 0\} = F$  .
  hence analytic_factor_p' h S K
    using ind ah KS con cm nh
  unfolding analytic_factor_p_def by auto
  then obtain g n and  $\alpha :: \text{nat} \Rightarrow \text{complex}$ 
  where ag: g analytic_on S and
    ng:  $\bigwedge z. z \in K \implies g z \neq 0$  and
    h_z:  $\bigwedge z. z \in S \implies h z = g z * (\prod_{k < n. z - \alpha k}$  and
    Im $\alpha$ :  $\alpha \text{ ' } \{.. < n\} \subseteq K$ 
  unfolding analytic_factor_p'_def by fastforce
  define  $\beta$  where  $\beta j \equiv \text{if } j < n \text{ then } \alpha j \text{ else } \xi$  for j
  show ?thesis
  unfolding analytic_factor_p'_def
  proof (intro ballI conjI exI)
    show g analytic_on S  $\bigwedge z. z \in K \implies g z \neq 0$ 
      by (rule ag, rule ng)
  next
    fix z assume *:  $z \in S$ 
    show  $f z = g z * (\prod_{j < n+k. z - \beta j}$ 
    proof -
      have  $(\prod_{j < n. z - \beta j} = (\prod_{j < n. z - \alpha j}$ 
         $(\prod_{j = n.. < n+k. z - \beta j} = (z - \xi) ^ k$ 
      unfolding  $\beta$ _def by auto
      moreover have  $(\prod_{j < n+k. z - \beta j} = (\prod_{j < n. z - \beta j} * (\prod_{j = n.. < n+k. z - \beta j}$ 
      by (metis Metric_Arith.nnf_simps(8) atLeast0LessThan
        not_add_less1 prod.atLeastLessThan_concat zero_order(1))
      ultimately have  $(\prod_{j < n+k. z - \beta j} = (z - \xi) ^ k * (\prod_{j < n. z - \alpha j}$  by auto

```

```

moreover have  $f z = g z * ((z - \xi) \wedge k * (\prod_{j < n} z - \alpha j))$ 
by (subst f_z; (subst h_z)?, use * in auto)
ultimately show ?thesis by auto
qed
next
show  $\beta \text{ ' } \{..<n + k\} \subseteq K$  unfolding  $\beta\_def$  using  $Im\alpha \langle \xi \in K \rangle$  by auto
qed
qed
qed

```

A nontrivial analytic function on connected compact region can be factorized as a everywhere-non-zero function and linear terms $z - s_0$ for all zeros s_0 . Note that the connected assumption of K may be removed, but we remain it just for simplicity of proof.

theorem *analytic_factorization:*

```

assumes af: f analytic_on S
and KS: K ⊆ S
and con: connected K
and compact K
and f not_zero_on K
obtains g n and  $\alpha :: nat \Rightarrow complex$  where
  g analytic_on S
   $\bigwedge z. z \in K \implies g z \neq 0$ 
   $\bigwedge z. z \in S \implies f z = g z * (\prod_{k < n} (z - \alpha k))$ 
   $\alpha \text{ ' } \{..<n\} \subseteq K$ 

```

proof –

```

have  $\langle finite \{z \in K. f z = 0\} \rangle$  using assms by (rule analytic_compact_finite_zeros)
moreover have  $\langle finite F \implies analytic\_factor\_p F \rangle$  for  $F$ 
by (induct rule: finite_induct; rule analytic_factorization_E analytic_factorization_I)
ultimately have analytic_factor_p  $\{z \in K. f z = 0\}$  by auto
hence analytic_factor_p' f S K unfolding analytic_factor_p_def using assms by auto
thus ?thesis unfolding analytic_factor_p'_def using assms that by metis

```

qed

3.3 Schwarz theorem in complex analysis

lemma *Schwarz_Lemma1:*

```

fixes  $f :: complex \Rightarrow complex$ 
and  $\xi :: complex$ 
assumes f holomorphic_on ball 0 1
and  $f 0 = 0$ 
and  $\bigwedge z. \|z\| < 1 \implies \|f z\| \leq 1$ 
and  $\|\xi\| < 1$ 
shows  $\|f \xi\| \leq \|\xi\|$ 
proof (cases f constant_on ball 0 1)
assume f constant_on ball 0 1
thus ?thesis unfolding constant_on_def
using assms by auto

```

next

```

assume nc: ¬f constant_on ball 0 1
have  $\bigwedge z. \|z\| < 1 \implies \|f z\| < 1$ 
proof –
fix  $z :: complex$  assume  $\|z\| < 1$ 
have  $\|f z\| \neq 1$ 
proof
assume  $\|f z\| = 1$ 
hence  $\bigwedge w. w \in ball 0 1 \implies \|f w\| \leq \|f z\|$ 

```

```

    using assms( $\beta$ ) by auto
  hence f constant_on ball 0 1
    by (intro maximum_modulus_principle [where  $U = \text{ball } 0 \ 1$  and  $\xi = z$ ])
      (use * assms(1) in auto)
  thus False using nc by blast
qed
with assms( $\beta$ ) [OF *] show  $\|f z\| < 1$  by auto
qed
thus  $\|f \xi\| \leq \|\xi\|$  by (intro Schwarz_Lemma(1), use assms in auto)
qed

```

theorem *Schwarz_Lemma2*:

```

fixes f :: complex  $\Rightarrow$  complex
and  $\xi$  :: complex
assumes holf: f holomorphic_on ball 0 R
and hR:  $0 < R$  and nz:  $f \ 0 = 0$ 
and bn:  $\bigwedge z. \|z\| < R \implies \|f z\| \leq 1$ 
and  $\xi R$ :  $\|\xi\| < R$ 
shows  $\|f \xi\| \leq \|\xi\| / R$ 
proof -
define  $\varphi$  where  $\varphi z \equiv f (R * z)$  for  $z :: \text{complex}$ 
have  $\|\xi / R\| < 1$  using  $\xi R$  hR by (subst nonzero_norm_divide, auto)
moreover have f holomorphic_on (*) ( $R :: \text{complex}$ ) ‘ball 0 1’
  by (rule holomorphic_on_subset, rule holf)
  (use hR in <auto simp: norm_mult>)
hence ( $f \circ (\lambda z. R * z)$ ) holomorphic_on ball 0 1
  by (auto intro: holomorphic_on_compose)
moreover have  $\varphi \ 0 = 0$  unfolding  $\varphi\_def$  using nz by auto
moreover have  $\bigwedge z. \|z\| < 1 \implies \|\varphi z\| \leq 1$ 
proof -
  fix  $z :: \text{complex}$  assume  $\|z\| < 1$ 
  have  $\|R * z\| < R$  using hR * by (fold scaleR_conv_of_real) auto
  thus  $\|\varphi z\| \leq 1$  unfolding  $\varphi\_def$  using bn by auto
qed
ultimately have  $\|\varphi (\xi / R)\| \leq \|\xi / R\|$ 
  unfolding comp_def by (fold  $\varphi\_def$ , intro Schwarz_Lemma1)
thus ?thesis unfolding  $\varphi\_def$  using hR by (subst (asm) nonzero_norm_divide, auto)
qed

```

3.4 Borel-Carathedory theorem

Borel-Carathedory theorem, from book *Theorem 5.5, The Theory of Functions, E. C. Titchmarsh*

lemma *Borel_Carathedory1*:

```

assumes hr:  $0 < R \ 0 < r \ r < R$ 
and f0:  $f \ 0 = 0$ 
and hf:  $\bigwedge z. \|z\| < R \implies \text{Re} (f z) \leq A$ 
and holf: f holomorphic_on (ball 0 R)
and zr:  $\|z\| \leq r$ 
shows  $\|f z\| \leq 2 * r / (R - r) * A$ 
proof -
have  $A\_ge\_0$ :  $A \geq 0$ 
using f0 hf by (metis hr(1) norm_zero zero_complex.simps(1))
then consider  $A = 0 \mid A > 0$  by linarith
thus  $\|f z\| \leq 2 * r / (R - r) * A$ 
proof (cases)

```

```

assume *:  $A = 0$ 
have 1:  $\bigwedge w. w \in \text{ball } 0 R \implies \|\exp (f w)\| \leq \|\exp (f 0)\|$  using hf f0 * by auto
have 2:  $\exp \circ f$  constant_on (ball 0 R)
  by (rule maximum_modulus_principle [where  $f = \exp \circ f$  and  $U = \text{ball } 0 R$ ])
    (use 1 hr(1) in ⟨auto intro: holomorphic_on_compose holf holomorphic_on_exp⟩)
have f constant_on (ball 0 R)
proof (rule classical)
  assume *:  $\neg f$  constant_on ball 0 R
  have open (f ' (ball 0 R))
    by (rule open_mapping_thm [where  $S = \text{ball } 0 R$ ], use holf * in auto)
  then obtain e where  $e > 0$  and cball 0 e  $\subseteq f'$  (ball 0 R)
    by (metis hr(1) f0 centre_in_ball imageI open_contains_cball)
  then obtain w
    where hw:  $w \in \text{ball } 0 R$  f w = e
    by (metis abs_of_nonneg imageE less_eq_real_def mem_cball_0 norm_of_real subset_eq)
  have exp e = exp (f w)
    using hw(2) by (fold exp_of_real) auto
  also have ... = exp (f 0)
    using hw(1) 2 hr(1) unfolding constant_on_def comp_def by auto
  also have ... = exp (0 :: real) by (subst f0) auto
  finally have e = 0 by auto
  with ⟨e > 0⟩ show ?thesis by blast
qed
hence f z = 0 using f0 hr zr unfolding constant_on_def by auto
hence  $\|f z\| = 0$  by auto
also have ...  $\leq 2 * r / (R - r) * A$  using hr ⟨A ≥ 0⟩ by auto
finally show ?thesis .
next
assume A_gt_0:  $A > 0$ 
define  $\varphi$  where  $\varphi z \equiv (f z) / (2 * A - f z)$  for  $z :: \text{complex}$ 
have  $\varphi$ _bound:  $\|\varphi z\| \leq 1$  if *:  $\|z\| < R$  for z
proof -
  define u v where  $u \equiv \text{Re } (f z)$  and  $v \equiv \text{Im } (f z)$ 
  hence  $u \leq A$  unfolding u_def using hf * by blast
  hence  $u^2 \leq (2 * A - u)^2$  using A_ge_0 by (simp add: sqrt_ge_absD)
  hence  $u^2 + v^2 \leq (2 * A - u)^2 + (-v)^2$  by auto
  moreover have  $2 * A - f z = \text{Complex } (2 * A - u) (-v)$  by (simp add: complex_eq_iff u_def v_def)
  hence  $\|f z\|^2 = u^2 + v^2$ 
     $\|2 * A - f z\|^2 = (2 * A - u)^2 + (-v)^2$ 
  unfolding u_def v_def using cmod_power2 complex.sel by presburger+
  ultimately have  $\|f z\|^2 \leq \|2 * A - f z\|^2$  by auto
  hence  $\|f z\| \leq \|2 * A - f z\|$  by auto
  thus ?thesis unfolding  $\varphi$ _def by (subst norm_divide) (simp add: divide_le_eq_1)
qed
moreover have nz:  $\bigwedge z :: \text{complex}. z \in \text{ball } 0 R \implies 2 * A - f z \neq 0$ 
proof
  fix z :: complex
  assume *:  $z \in \text{ball } 0 R$ 
    and eq:  $2 * A - f z = 0$ 
  hence  $\text{Re } (f z) \leq A$  using hf by auto
  moreover have  $\text{Re } (f z) = 2 * A$ 
    by (metis eq_Re_complex_of_real right_minus_eq)
  ultimately show False using A_gt_0 by auto
qed
ultimately have  $\varphi$  holomorphic_on ball 0 R

```

```

  unfolding  $\varphi\_def$   $comp\_def$  by (intro holomorphic_intros holf)
  moreover have  $\varphi 0 = 0$  unfolding  $\varphi\_def$  using  $f0$  by auto
  ultimately have *:  $\|\varphi z\| \leq \|z\| / R$ 
  using  $hr(1)$   $\varphi\_bound$   $zr$   $hr$  Schwarz_Lemma2 by auto
  also have ...  $< 1$  using  $zr$   $hr$  by auto
  finally have  $h\varphi$ :  $\|\varphi z\| \leq r / R$   $\|\varphi z\| < 1$   $1 + \varphi z \neq 0$ 
  proof (safe)
    show  $\|\varphi z\| \leq r / R$  using *  $zr$   $hr(1)$ 
    by (metis divide_le_cancel dual_order.trans nle_le)
  next
  assume  $1 + \varphi z = 0$ 
  hence  $\varphi z = -1$  using add_eq_0_iff by blast
  thus  $\|\varphi z\| < 1 \implies False$  by auto
  qed
  have  $2*A - fz \neq 0$  using  $nz$   $hr(3)$   $zr$  by auto
  hence  $fz = 2*A*\varphi z / (1 + \varphi z)$ 
  using  $h\varphi(3)$  unfolding  $\varphi\_def$  by (auto simp add: field_simps)
  hence  $\|fz\| = 2*A*\|\varphi z\| / \|1 + \varphi z\|$ 
  by (auto simp add: norm_divide norm_mult A_ge_0)
  also have ...  $\leq 2*A*(\|\varphi z\| / (1 - \|\varphi z\|))$ 
  proof -
    have  $\|1 + \varphi z\| \geq 1 - \|\varphi z\|$ 
    by (metis norm_diff_ineq norm_one)
    thus ?thesis
    by (simp, rule divide_left_mono, use A_ge_0 in auto)
    (intro mult_pos_pos, use  $h\varphi(2)$  in auto)
  qed
  also have ...  $\leq 2*A*((r/R) / (1 - r/R))$ 
  proof -
    have *:  $a / (1 - a) \leq b / (1 - b)$ 
    if  $a < 1$   $b < 1$   $a \leq b$  for  $a b :: real$ 
    using that by (auto simp add: field_simps)
    have  $\|\varphi z\| / (1 - \|\varphi z\|) \leq (r/R) / (1 - r/R)$ 
    by (rule *; (intro  $h\varphi$ )?) (use  $hr$  in auto)
    thus ?thesis by (rule mult_left_mono, use A_ge_0 in auto)
  qed
  also have ...  $= 2*r/(R-r) * A$  using  $hr(1)$  by (auto simp add: field_simps)
  finally show ?thesis .
  qed
  qed

```

lemma Borel_Caratheodory2:

```

  assumes  $hr$ :  $0 < R$   $0 < r$   $r < R$ 
  and  $hf$ :  $\bigwedge z. \|z\| < R \implies Re(fz - f0) \leq A$ 
  and  $holf$ :  $f$  holomorphic_on (ball 0 R)
  and  $zr$ :  $\|z\| \leq r$ 
  shows  $\|fz - f0\| \leq 2*r/(R-r) * A$ 
  proof -
    define  $g$  where  $g z \equiv fz - f0$  for  $z$ 
    show ?thesis
    by (fold  $g\_def$ , rule Borel_Caratheodory1)
    (unfold  $g\_def$ , insert  $assms$ , auto intro: holomorphic_intros)
  qed

```

theorem Borel_Caratheodory3:

```

assumes hr:  $0 < R$   $0 < r$   $r < R$ 
and hf:  $\bigwedge w. w \in \text{ball } s \ R \implies \text{Re } (f \ w - f \ s) \leq A$ 
and holf:  $f \ \text{holomorphic\_on } (\text{ball } s \ R)$ 
and zr:  $z \in \text{ball } s \ r$ 
shows  $\|f \ z - f \ s\| \leq 2 * r / (R - r) * A$ 
proof -
define g where  $g \ w \equiv f \ (s + w)$  for w
have  $\bigwedge w. \|w\| < R \implies \text{Re } (f \ (s + w) - f \ s) \leq A$ 
by (intro hf) (auto simp add: dist_complex_def)
hence  $\|g \ (z - s) - g \ 0\| \leq 2 * r / (R - r) * A$ 
by (intro Borel_Caratheodory2, unfold g_def, insert asms)
      (auto intro: holomorphic_intros simp add: dist_complex_def norm_minus_commute)
thus ?thesis unfolding g_def by auto
qed

```

3.5 Lemma 3.9

These lemmas is referred to the following material: Theorem 3.9, *The Theory of the Riemann Zeta-Function*, E. C. Titchmarsh, D. R. Heath-Brown.

```

lemma lemma_3_9_beta1:
fixes f M r s0
assumes zl:  $0 < r$   $0 \leq M$ 
and hf:  $f \ \text{holomorphic\_on } \text{ball } 0 \ r$ 
and ne:  $\bigwedge z. z \in \text{ball } 0 \ r \implies f \ z \neq 0$ 
and bn:  $\bigwedge z. z \in \text{ball } 0 \ r \implies \|f \ z / f \ 0\| \leq \exp \ M$ 
shows  $\|\logderiv \ f \ 0\| \leq 4 * M / r$ 
and  $\forall s \in \text{cball } 0 \ (r / 4). \|\logderiv \ f \ s\| \leq 8 * M / r$ 
proof (goal_cases)
obtain g
where holg:  $g \ \text{holomorphic\_on } \text{ball } 0 \ r$ 
and exp_g:  $\bigwedge x. x \in \text{ball } 0 \ r \implies \exp \ (g \ x) = f \ x$ 
by (rule holomorphic_logarithm_exists [of ball 0 r f 0])
      (use zl(1) ne hf in auto)
have f0:  $\exp \ (g \ 0) = f \ 0$  using exp_g zl(1) by auto
have Re (g z - g 0)  $\leq M$  if *:  $\|z\| < r$  for z
proof -
have  $\exp \ (\text{Re } (g \ z - g \ 0)) = \|\exp \ (g \ z - g \ 0)\|$ 
by (rule norm_exp_eq_Re [symmetric])
also have ... =  $\|f \ z / f \ 0\|$ 
by (subst exp_diff, subst f0, subst exp_g)
      (use * in auto)
also have ...  $\leq \exp \ M$  by (rule bn) (use * in auto)
finally show ?thesis by auto
qed
hence  $\|g \ z - g \ 0\| \leq 2 * (r / 2) / (r - r / 2) * M$ 
if *:  $\|z\| \leq r / 2$  for z
by (intro Borel_Caratheodory2 [where f = g])
      (use zl(1) holg * in auto)
also have ... =  $2 * M$  using zl(1) by auto
finally have hg:  $\bigwedge z. \|z\| \leq r / 2 \implies \|g \ z - g \ 0\| \leq 2 * M$  .
have result:  $\|\logderiv \ f \ s\| \leq 2 * M / r'$ 
when cball s r'  $\subseteq$  cball 0 (r / 2)  $0 < r'$   $\|s\| < r / 2$  for s r'
proof -
have contain:  $\bigwedge z. \|s - z\| \leq r' \implies \|z\| \leq r / 2$ 
using that by (auto simp add: cball_def subset_eq dist_complex_def)

```

have *contain'*: $\|z\| < r$ **when** $\|s - z\| \leq r'$ **for** z
using *zl(1) contain [of z] that by auto*
have *s_in_ball*: $s \in \text{ball } 0 \ r$ **using** *that(3) zl(1) by auto*
have *deriv f s = deriv (λx. exp (g x)) s*
by (*rule deriv_cong_ev, subst eventually_nhds*)
(*rule exI [where x = ball 0 (r / 2)], use exp_g zl(1) that(3) in auto*)
also have $\dots = \text{exp } (g \ s) * \text{deriv } g \ s$
by (*intro DERIV_fun_exp [THEN DERIV_imp_deriv] field_differentiable_derivI*)
(*meson holg open_ball s_in_ball holomorphic_on_imp_differentiable_at*)
finally have *df: logderiv f s = deriv g s*
proof –
assume *deriv f s = exp (g s) * deriv g s*
moreover have $f \ s \neq 0$ **by** (*intro ne s_in_ball*)
ultimately show *?thesis*
unfolding *logderiv_def using exp_g [OF s_in_ball] by auto*
qed
have $\bigwedge z. \|s - z\| = r' \implies \|g \ z - g \ 0\| \leq 2 * M$
using *contain by (intro hg) auto*
moreover have $(\lambda z. g \ z - g \ 0)$ *holomorphic_on cball s r'*
by (*rule holomorphic_on_subset [where s=ball 0 r], insert holg*)
(*auto intro: holomorphic_intros contain' simp add: dist_complex_def*)
moreover hence *continuous_on (cball s r')* $(\lambda z. g \ z - g \ 0)$
by (*rule holomorphic_on_imp_continuous_on*)
ultimately have $\|(\text{deriv } \sim 1) (\lambda z. g \ z - g \ 0) \ s\| \leq \text{fact } 1 * (2 * M) / r' \wedge 1$
using *that(2) by (intro Cauchy_inequality) auto*
also have $\dots = 2 * M / r'$ **by** *auto*
also have *deriv g s = deriv (λz. g z - g 0) s*
by (*subst deriv_diff, auto*)
(*rule holomorphic_on_imp_differentiable_at, use holg s_in_ball in auto*)
hence $\| \text{deriv } g \ s \| = \| (\text{deriv } \sim 1) (\lambda z. g \ z - g \ 0) \ s \|$
by (*auto simp add: derivative_intros*)
ultimately show *?thesis by (subst df) auto*
qed
case 1 show *?case using result [of 0 r / 2] zl(1) by auto*
case 2 show *?case proof safe*
fix $s :: \text{complex}$ **assume** *hs: s ∈ cball 0 (r / 4)*
hence $z \in \text{cball } s \ (r / 4) \implies \|z\| \leq r / 2$ **for** z
using *norm_triangle_sub [of z s]*
by (*auto simp add: dist_complex_def norm_minus_commute*)
hence $\| \text{logderiv } f \ s \| \leq 2 * M / (r / 4)$
by (*intro result*) (*use zl(1) hs in auto*)
also have $\dots = 8 * M / r$ **by** *auto*
finally show $\| \text{logderiv } f \ s \| \leq 8 * M / r$.
qed
qed

lemma *lemma_3_9_beta1'*:

fixes $f \ M \ r \ s_0$
assumes *zl: 0 < r 0 ≤ M*
and *hf: f holomorphic_on ball s r*
and *ne: λz. z ∈ ball s r ⇒ f z ≠ 0*
and *bn: λz. z ∈ ball s r ⇒ \|f z / f s\| ≤ exp M*
and *hs: z ∈ cball s (r / 4)*
shows $\| \text{logderiv } f \ z \| \leq 8 * M / r$

proof –


```

define  $g$  where  $g\ z \equiv f\ (s + z)$  for  $z$ 
have  $\forall z \in \text{cball } 0\ (r / 4). \|\text{logderiv } g\ z\| \leq 8 * M / r$ 
  by (intro lemma_3_9_beta1 assms, unfold  $g\_def$ )
    (auto simp add: dist_complex_def intro!: assms holomorphic_on_shift)
note  $bspec$  [OF this, of  $z - s$ ]
moreover have  $f$  field_differentiable at  $z$ 
  by (rule holomorphic_on_imp_differentiable_at [where ?s = ball s r])
    (insert hs zl(1), auto intro: hf simp add: dist_complex_def)
ultimately show ?thesis unfolding  $g\_def$  using  $hs$ 
  by (auto simp add: dist_complex_def logderiv_shift)
qed

```

lemma lemma_3_9_beta2:

```

fixes  $f\ M\ r$ 
assumes  $zl: 0 < r\ 0 \leq M$ 
  and  $af: f$  analytic_on cball 0  $r$ 
  and  $f0: f\ 0 \neq 0$ 
  and  $rz: \bigwedge z. z \in \text{cball } 0\ r \implies \text{Re } z > 0 \implies f\ z \neq 0$ 
  and  $bn: \bigwedge z. z \in \text{cball } 0\ r \implies \|f\ z / f\ 0\| \leq \text{exp } M$ 
  and  $hg: \Gamma \subseteq \{z \in \text{cball } 0\ (r / 2). f\ z = 0 \wedge \text{Re } z \leq 0\}$ 
shows  $-\text{Re } (\text{logderiv } f\ 0) \leq 8 * M / r + \text{Re } (\sum_{z \in \Gamma}. 1 / z)$ 

```

proof –

```

have  $nz': f$  not_zero_on cball 0  $(r / 2)$ 
  unfolding not_zero_on_def using  $f0\ zl(1)$  by auto
hence  $fin\_zeros: \text{finite } \{z \in \text{cball } 0\ (r / 2). f\ z = 0\}$ 
  by (intro analytic_compact_finite_zeros [where  $S = \text{cball } 0\ r$ ])
    (use  $af\ zl$  in auto)
obtain  $g\ n$  and  $\alpha :: \text{nat} \Rightarrow \text{complex}$ 
where  $ag: g$  analytic_on cball 0  $r$ 
  and  $ng: \bigwedge z. z \in \text{cball } 0\ (r / 2) \implies g\ z \neq 0$ 
  and  $fac: \bigwedge z. z \in \text{cball } 0\ r \implies f\ z = g\ z * (\prod_{k < n}. (z - \alpha\ k))$ 
  and  $\text{Im } \alpha: \alpha\ \{.. < n\} \subseteq \text{cball } 0\ (r / 2)$ 
  by (rule analytic_factorization [
    where  $K = \text{cball } 0\ (r / 2)$ 
    and  $S = \text{cball } 0\ r$  and  $f = f$ ])
    (use  $zl(1)\ af\ nz'$  in auto)
have  $g0: \|g\ 0\| \neq 0$  using  $ng\ zl(1)$  by auto
hence  $g$  holomorphic_on cball 0  $r$ 
  ( $\lambda z. g\ z / g\ 0$ ) holomorphic_on cball 0  $r$ 
  using  $ag$  by (auto simp add: analytic_intros intro: analytic_imp_holomorphic)
hence  $holg:$ 
   $g$  holomorphic_on ball 0  $r$ 
  ( $\lambda z. g\ z / g\ 0$ ) holomorphic_on ball 0  $r$ 
  continuous_on (cball 0  $r$ ) ( $\lambda z. g\ z / g\ 0$ )
  by (auto intro!: holomorphic_on_imp_continuous_on
    holomorphic_on_subset [where  $t = \text{ball } 0\ r$ ])
have  $nz\_alpha: \bigwedge k. k < n \implies \alpha\ k \neq 0$  using  $zl(1)\ f0\ fac$  by auto
have  $\|g\ z / g\ 0\| \leq \text{exp } M$  if  $*$ :  $z \in \text{sphere } 0\ r$  for  $z$ 
proof –
  let  $?p = \|(\prod_{k < n}. (z - \alpha\ k)) / (\prod_{k < n}. (0 - \alpha\ k))\|$ 
  have 1:  $\|f\ z / f\ 0\| \leq \text{exp } M$  using  $bn\ *$  by auto
  have 2:  $\|f\ z / f\ 0\| = \|g\ z / g\ 0\| * ?p$ 
  by (subst norm_mult [symmetric], subst (1 2)  $fac$ )
    (use that  $zl(1)$  in auto)
  have  $?p = (\prod_{k < n}. (\|z - \alpha\ k\| / \|0 - \alpha\ k\|))$ 

```

by (auto simp add: prod_norm [symmetric] norm_divide prod_dividef)
 also have $\|z - \alpha k\| \geq \|0 - \alpha k\|$ if $k < n$ for k
 proof (rule ccontr)
 assume **: $\neg \|z - \alpha k\| \geq \|0 - \alpha k\|$
 have $r = \|z\|$ using * by auto
 also have $\dots \leq \|0 - \alpha k\| + \|z - \alpha k\|$ by (simp add: norm_triangle_sub)
 also have $\dots < 2 * \|\alpha k\|$ using ** by auto
 also have $\alpha k \in \text{cball } 0 (r / 2)$ using $\text{Im } \alpha$ that by blast
 hence $2 * \|\alpha k\| \leq r$ by auto
 finally show *False* by linarith
 qed
 hence $\bigwedge k. k < n \implies \|z - \alpha k\| / \|0 - \alpha k\| \geq 1$
 using $\text{nz_}\alpha$ by (subst le_divide_eq_1_pos) auto
 hence $(\prod_{k < n}. (\|z - \alpha k\| / \|0 - \alpha k\|)) \geq 1$ by (rule prod_ge_1) simp
 finally have $\exists: ?p \geq 1$.
 have rule1: $b = a * c \implies a \geq 0 \implies c \geq 1 \implies a \leq b$ for $a b c :: \text{real}$
 by (metis landau_omega.R_mult_left_mono more_arith_simps(6))
 have $\|g z / g 0\| \leq \|f z / f 0\|$
 by (rule rule1) (rule 2 3 norm_ge_zero)+
 thus ?thesis using 1 by linarith
 qed
 hence $\bigwedge z. z \in \text{cball } 0 r \implies \|g z / g 0\| \leq \exp M$
 using holg
 by (auto intro: maximum_modulus_frontier
 [where $f = \lambda z. g z / g 0$ and $S = \text{cball } 0 r$])
 hence bn' : $\bigwedge z. z \in \text{cball } 0 (r / 2) \implies \|g z / g 0\| \leq \exp M$ using $\text{zl}(1)$ by auto
 have ag' : g analytic_on $\text{cball } 0 (r / 2)$
 by (rule analytic_on_subset [where $S = \text{cball } 0 r$])
 (use $\text{ag } \text{zl}(1)$ in auto)
 have $\|\logderiv g 0\| \leq 4 * M / (r / 2)$
 by (rule lemma_3_9_beta1(1) [where $f = g$])
 (use $\text{zl ng bn}' \text{holg}$ in auto)
 also have $\dots = 8 * M / r$ by auto
 finally have bn_g : $\|\logderiv g 0\| \leq 8 * M / r$ unfolding \logderiv_def by auto
 let $?P = \lambda w. \prod_{k < n}. (w - \alpha k)$
 let $?S' = \sum_{k < n}. \logderiv (\lambda z. z - \alpha k) 0$
 let $?S = \sum_{k < n}. -(1 / \alpha k)$
 have g field_differentiable at 0 using holg $\text{zl}(1)$
 by (auto intro!: holomorphic_on_imp_differentiable_at)
 hence ld_g : g log_differentiable 0 unfolding $\log_differentiable_def$ using $g0$ by auto
 have $\logderiv ?P 0 = ?S'$ and ld_P : $?P$ log_differentiable 0
 by (auto intro!: $\logderiv_linear \text{nz_}\alpha \logderiv_prod$)
 note this(1)
 also have $?S' = ?S$
 by (rule sum.cong)
 (use $\text{nz_}\alpha$ in auto cong: $\logderiv_linear(1)$)
 finally have cd_P : $\logderiv ?P 0 = ?S$.
 have $\logderiv f 0 = \logderiv (\lambda z. g z * ?P z) 0$
 by (rule \logderiv_cong_ev , subst eventually_nhds)
 (intro exI [where $x = \text{ball } 0 r$], use $\text{fac } \text{zl}(1)$ in auto)
 also have $\dots = \logderiv g 0 + \logderiv ?P 0$
 by (subst \logderiv_mult) (use $\text{ld}_g \text{ld}_P$ in auto)
 also have $\dots = \logderiv g 0 + ?S$ using cd_P by auto
 finally have $\text{Re } (\logderiv f 0) = \text{Re } (\logderiv g 0) + \text{Re } ?S$ by simp
 moreover have $-\text{Re } (\sum_{z \in \Gamma}. 1 / z) \leq \text{Re } ?S$

proof –

have – $Re (\sum z \in \Gamma. 1 / z) = (\sum z \in \Gamma. Re (- (1 / z)))$ **by** *(auto simp add: sum_negf)*

also have $\dots \leq (\sum k < n. Re (- (1 / \alpha k)))$

proof *(rule sum_le_included)*

show $\forall z \in \Gamma. \exists k \in \{..<n\}. \alpha k = z \wedge Re (- (1 / z)) \leq Re (- (1 / \alpha k))$
(is Ball _ ?P)

proof

fix z **assume** $hz: z \in \Gamma$

have $\exists k \in \{..<n\}. \alpha k = z$

proof *(rule ccontr)*

assume $ne_\alpha: \neg (\exists k \in \{..<n\}. \alpha k = z)$

have $z_in: z \in cball\ 0\ (r / 2) \wedge z \in cball\ 0\ r$ **using** $hg\ hz\ zl(1)$ **by** *auto*

hence $g\ z \neq 0$ **using** ng **by** *auto*

moreover have $(\prod k < n. (z - \alpha k)) \neq 0$ **using** $ne_\alpha\ hz$ **by** *auto*

ultimately have $f\ z \neq 0$ **using** $fac\ z_in$ **by** *auto*

moreover have $f\ z = 0$ **using** $hz\ hg$ **by** *auto*

ultimately show *False* **by** *auto*

qed

thus $?P\ z$ **by** *auto*

qed

show $\forall k \in \{..<n\}. 0 \leq Re (- (1 / \alpha k))$ *(is Ball _ ?P)*

proof

fix k **assume** $*$: $k \in \{..<n\}$

have $1: \alpha k \in cball\ 0\ r$ **using** $Im\ \alpha\ zl(1)\ *$ **by** *auto*

hence $(\prod j < n. (\alpha k - \alpha j)) = 0$

by *(subst prod_zero_iff) (use * in auto)*

with 1 **have** $f\ (\alpha k) = 0$ **by** *(subst fac) auto*

hence $Re\ (\alpha k) \leq 0$ **using** $1\ rz\ f0$ **by** *fastforce*

hence $Re\ (1 * cnj\ (\alpha k)) \leq 0$ **by** *auto*

thus $?P\ k$ **using** $Re_complex_div_le_0$ **by** *auto*

qed

show *finite* $\{..<n\}$ **by** *auto*

have $\Gamma \subseteq \{z \in cball\ 0\ (r / 2). f\ z = 0\}$ **using** hg **by** *auto*

thus *finite* Γ **using** fin_zeros **by** *(rule finite_subset)*

qed

also have $\dots = Re\ ?S$ **by** *auto*

finally show *?thesis* .

qed

ultimately have $- Re\ (logderiv\ f\ 0) - Re\ (\sum z \in \Gamma. 1 / z) \leq Re\ (- logderiv\ g\ 0)$ **by** *auto*

also have $\dots \leq \|- logderiv\ g\ 0\|$ **by** *(rule complex_Re_le_cmod)*

also have $\dots \leq 8 * M / r$ **by** *simp (rule bn_g)*

finally show *?thesis* **by** *auto*

qed

theorem *lemma_3_9_beta3*:

fixes $f\ M\ r$ **and** $s :: complex$

assumes $zl: 0 < r \wedge 0 \leq M$

and $af: f$ *analytic_on* $cball\ s\ r$

and $f0: f\ s \neq 0$

and $rz: \bigwedge z. z \in cball\ s\ r \implies Re\ z > Re\ s \implies f\ z \neq 0$

and $bn: \bigwedge z. z \in cball\ s\ r \implies \|f\ z / f\ s\| \leq exp\ M$

and $hg: \Gamma \subseteq \{z \in cball\ s\ (r / 2). f\ z = 0 \wedge Re\ z \leq Re\ s\}$

shows $- Re\ (logderiv\ f\ s) \leq 8 * M / r + Re\ (\sum z \in \Gamma. 1 / (z - s))$

proof –

define g **where** $g \equiv f \circ (\lambda z. s + z)$

```

define Δ where Δ ≡ (λz. z - s) ‘ Γ
hence 1: g analytic_on cball 0 r
  unfolding g_def using af
  by (intro analytic_on_compose) (auto simp add: analytic_intros)
moreover have g 0 ≠ 0 unfolding g_def using f0 by auto
moreover have (Re z > 0 → g z ≠ 0) ∧ ‖g z / g 0‖ ≤ exp M
  if hz: z ∈ cball 0 r for z
proof (intro impI conjI)
  assume hz': 0 < Re z
  thus g z ≠ 0 unfolding g_def comp_def
    using hz by (intro rz) (auto simp add: dist_complex_def)
next
  show ‖g z / g 0‖ ≤ exp M
    unfolding g_def comp_def using hz
    by (auto simp add: dist_complex_def intro!: bn)
qed
moreover have Δ ⊆ {z ∈ cball 0 (r / 2). g z = 0 ∧ Re z ≤ 0}
proof safe
  fix z assume z ∈ Δ
  hence s + z ∈ Γ unfolding Δ_def by auto
  thus g z = 0 Re z ≤ 0 z ∈ cball 0 (r / 2)
    unfolding g_def comp_def using hg by (auto simp add: dist_complex_def)
qed
ultimately have - Re (logderiv g 0) ≤ 8 * M / r + Re (∑ z∈Δ. 1 / z)
  by (intro lemma_3_9_beta2) (use zl in auto)
also have ... = 8 * M / r + Re (∑ z∈Γ. 1 / (z - s))
  unfolding Δ_def by (subst sum.reindex) (unfold inj_on_def, auto)
finally show ?thesis
  unfolding g_def comp_def using zl(1)
  by (subst (asm) logderiv_shift)
    (auto intro: analytic_on_imp_differentiable_at [OF af])
qed

```

```

unbundle no_pnt_notation
end
theory Zeta_Zerofree
imports
  PNT_Complex_Analysis_Lemmas
begin
unbundle pnt_notation

```

4 Zero-free region of zeta function

```

lemma cos_inequality_1:
  fixes x :: real
  shows 3 + 4 * cos x + cos (2 * x) ≥ 0
proof -
  have cos (2 * x) = (cos x)2 - (sin x)2
    by (rule cos_double)
  also have ... = (cos x)2 - (1 - (cos x)2)
    unfolding sin_squared_eq ..
  also have ... = 2 * (cos x)2 - 1 by auto
  finally have 1: cos (2 * x) = 2 * (cos x)2 - 1 .
  have 0 ≤ 2 * (1 + cos x)2 by auto
  also have ... = 3 + 4 * cos x + (2 * (cos x)2 - 1)

```

by (simp add: field_simps power2_eq_square)
 finally show ?thesis unfolding 1.

qed

lemma multiplicative_fds_zeta:

completely_multiplicative_function (fds_nth fds_zeta_complex)
 by standard auto

lemma fds_mangoldt_eq:

fds_mangoldt_complex = -(fds_deriv fds_zeta / fds_zeta)

proof -

have fds_nth fds_zeta_complex 1 \neq 0 by auto

hence fds_nth (fds_deriv fds_zeta_complex / fds_zeta) n = -fds_nth fds_zeta n * mangoldt n for n

using multiplicative_fds_zeta

by (intro fds_nth_logderiv_completely_multiplicative)

thus ?thesis by (intro fds_eqI, auto)

qed

lemma abs_conv_abscissa_log_deriv:

abs_conv_abscissa (fds_deriv fds_zeta_complex / fds_zeta) \leq 1

by (rule abs_conv_abscissa_completely_multiplicative_log_deriv
 [OF multiplicative_fds_zeta, unfolded abs_conv_abscissa_zeta], auto)

lemma abs_conv_abscissa_mangoldt:

abs_conv_abscissa (fds_mangoldt_complex) \leq 1

using abs_conv_abscissa_log_deriv

by (subst fds_mangoldt_eq, subst abs_conv_abscissa_minus)

lemma

assumes s: $\text{Re } s > 1$

shows eval_fds_mangoldt: eval_fds (fds_mangoldt) s = - deriv zeta s / zeta s

and abs_conv_mangoldt: fds_abs_converges (fds_mangoldt) s

proof -

from abs_conv_abscissa_log_deriv

have 1: abs_conv_abscissa (fds_deriv fds_zeta_complex / fds_zeta) $<$ ereal (s * 1)

using s by (intro le_ereal_less, auto simp: one_ereal_def)

have 2: abs_conv_abscissa fds_zeta_complex $<$ ereal (s * 1)

using s by (subst abs_conv_abscissa_zeta, auto)

hence 3: fds_abs_converges (fds_deriv fds_zeta_complex / fds_zeta) s

by (intro fds_abs_converges) (rule 1)

have eval_fds (fds_mangoldt) s = eval_fds (-(fds_deriv fds_zeta_complex / fds_zeta)) s

using fds_mangoldt_eq by auto

also have ... = -eval_fds (fds_deriv fds_zeta_complex / fds_zeta) s

by (intro eval_fds_uminus fds_abs_converges_imp_converges 3)

also have ... = -(eval_fds (fds_deriv fds_zeta_complex) s / eval_fds fds_zeta s)

using s by (subst eval_fds_log_deriv; ((intro 1 2)?, (auto intro!: eval_fds_zeta_nonzero)?))

also have ... = - deriv zeta s / zeta s

using s by (subst eval_fds_zeta, blast, subst eval_fds_deriv_zeta, auto)

finally show eval_fds (fds_mangoldt) s = - deriv zeta s / zeta s .

show fds_abs_converges (fds_mangoldt) s

by (subst fds_mangoldt_eq) (intro fds_abs_converges_uminus 3)

qed

lemma sums_mangoldt:

fixes s :: complex

assumes $s: \text{Re } s > 1$
shows $((\lambda n. \text{mangoldt } n / n \text{ nat_powr } s) \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s) \{1..\}$
proof –
let $?f = (\lambda n. \text{mangoldt } n / n \text{ nat_powr } s)$
have 1: $\text{fds_abs_converges } (\text{fds mangoldt}) s$
by $(\text{intro abs_conv_mangoldt } s)$
hence 2: $\text{fds_converges } (\text{fds mangoldt}) s$
by $(\text{rule fds_abs_converges_imp_converges})$
hence $\text{summable } (\lambda n. \|\text{fds_nth } (\text{fds mangoldt}) n / \text{nat_power } n s\|)$
by $(\text{fold fds_abs_converges_def, intro 1})$
moreover have $(\lambda n. \text{fds_nth } (\text{fds mangoldt}) n / \text{nat_power } n s) \text{ sums } (- \text{deriv zeta } s / \text{zeta } s)$
by $(\text{subst eval_fds_mangoldt}(1) [\text{symmetric}], \text{intro } s, \text{fold fds_converges_iff, intro 2})$
ultimately have $((\lambda n. \text{fds_nth } (\text{fds mangoldt}) n / n \text{ nat_powr } s) \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s)$
UNIV
by $(\text{fold nat_power_complex_def, rule norm_summable_imp_has_sum})$
moreover have $[\text{simp}]: (\text{if } n = 0 \text{ then } 0 \text{ else mangoldt } n) = \text{mangoldt } n$ **for** n **by** auto
ultimately have $(?f \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s) \text{ UNIV}$ **by** $(\text{auto simp add: fds_nth_fds})$
hence 3: $(?f \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s) \text{ UNIV}$ **by** auto
have $\text{sum } ?f \{0\} = 0$ **by** auto
moreover have $(?f \text{ has_sum } \text{sum } ?f \{0\}) \{0\}$
by $(\text{rule has_sum_finite, auto})$
ultimately have $(?f \text{ has_sum } 0) \{0\}$ **by** auto
hence $(?f \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s - 0) (\text{UNIV} - \{0\})$
by $(\text{intro has_sum_Diff } 3, \text{auto})$
moreover have $\text{UNIV} - \{0 :: \text{nat}\} = \{1..\}$ **by** auto
ultimately show $(?f \text{ has_sum } - \text{deriv zeta } s / \text{zeta } s) \{1..\}$ **by** auto
qed

lemma $\text{sums_Re_logderiv_zeta}$:

fixes $\sigma t :: \text{real}$
assumes $s: \sigma > 1$
shows $((\lambda n. \text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (t * \ln n)) \text{ has_sum } \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma t) / \text{zeta } (\text{Complex } \sigma t))) \{1..\}$
proof –
have $((\lambda x. \text{Re } (\text{mangoldt_complex } x / x \text{ nat_powr } \text{Complex } \sigma t)) \text{ has_sum } \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma t) / \text{zeta } (\text{Complex } \sigma t))) \{1..\}$
using s **by** $(\text{intro has_sum_Re sums_mangoldt}) \text{auto}$
moreover have $\text{Re } (\text{mangoldt } n / n \text{ nat_powr } (\text{Complex } \sigma t)) = \text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (t * \ln n)$ **if** $*$: $1 \leq n$ **for** n
proof –
let $?n = n :: \text{complex}$
have $1 / n \text{ nat_powr } (\text{Complex } \sigma t) = n \text{ nat_powr } (\text{Complex } (-\sigma) (-t))$
by $(\text{fold powr_minus_divide, auto simp add: legacy_Complex_simps})$
also have $\dots = \exp (\text{Complex } (-\sigma * \ln n) (-t * \ln n))$
unfolding powr_def **by** $(\text{auto simp add: field_simps legacy_Complex_simps, use } * \text{ in linarith})$
finally have $\text{Re } (1 / n \text{ nat_powr } (\text{Complex } \sigma t)) = \text{Re } \dots$ **by** auto
also have $\dots = n \text{ nat_powr } (-\sigma) * \cos (t * \ln n)$
by $(\text{unfold powr_def, subst Re_exp, use } * \text{ in auto})$
finally have 1: $\text{mangoldt_real } n * \text{Re } (1 / n \text{ nat_powr } (\text{Complex } \sigma t)) = \text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (t * \ln n)$ **by** auto
have $\text{rule_1}: \text{Re } (w * z) = \text{Re } w * \text{Re } z$ **if** $*$: $\text{Im } w = 0$ **for** $z w :: \text{complex}$ **using** $*$ **by** auto
have $\text{Re } (\text{mangoldt } n * (1 / n \text{ nat_powr } (\text{Complex } \sigma t))) = \text{mangoldt_real } n * \text{Re } (1 / n \text{ nat_powr } (\text{Complex } \sigma t))$
by $(\text{subst rule_1, auto})$
with 1 **show** $?thesis$ **by** auto

qed
ultimately show $((\lambda n. \text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (t * \ln (\text{real } n)))$
 $\text{has_sum } \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma t) / \text{zeta } (\text{Complex } \sigma t)) \{1..\}$
by $(\text{subst has_sum_cong}) \text{ auto}$
qed

lemma *logderiv_zeta_ineq*:
fixes $\sigma t :: \text{real}$
assumes $s: \sigma > 1$
shows $3 * \text{Re } (\text{logderiv zeta } (\text{Complex } \sigma 0)) + 4 * \text{Re } (\text{logderiv zeta } (\text{Complex } \sigma t))$
 $+ \text{Re } (\text{logderiv zeta } (\text{Complex } \sigma (2*t))) \leq 0$ (is $?x \leq 0$)
proof –
have $[\text{simp}]: \text{Re } (-z) = - \text{Re } z$ for z by *auto*
have $((\lambda n.$
 $3 * (\text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (0 * \ln n))$
 $+ 4 * (\text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (t * \ln n))$
 $+ 1 * (\text{mangoldt_real } n * n \text{ nat_powr } (-\sigma) * \cos (2*t * \ln n))$
 $) \text{ has_sum}$
 $3 * \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma 0) / \text{zeta } (\text{Complex } \sigma 0))$
 $+ 4 * \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma t) / \text{zeta } (\text{Complex } \sigma t))$
 $+ 1 * \text{Re } (- \text{deriv zeta } (\text{Complex } \sigma (2*t)) / \text{zeta } (\text{Complex } \sigma (2*t)))$
 $) \{1..\}$
by $(\text{intro has_sum_add has_sum_cmult_right sums_Re_logderiv_zeta } s)$
hence $*$: $((\lambda n. \text{mangoldt_real } n * n \text{ nat_powr } (-\sigma)$
 $* (3 + 4 * \cos (t * \ln n) + \cos (2 * (t * \ln n)))$
 $) \text{ has_sum } -?x) \{1..\}$
unfolding *logderiv_def* by $(\text{auto simp add: field_simps})$
have $-?x \geq 0$
by $(\text{rule has_sum_nonneg, rule } *,$
 $\text{intro mult_nonneg_nonneg,}$
 $\text{auto intro: mangoldt_nonneg cos_inequality_1})$
thus $?x \leq 0$ by *linarith*
qed

lemma *sums_zeta_real*:
fixes $r :: \text{real}$
assumes $1 < r$
shows $(\sum n. (n_+) \text{ powr } -r) = \text{Re } (\text{zeta } r)$
proof –
have $(\sum n. (n_+) \text{ powr } -r) = (\sum n. \text{Re } (n_+ \text{ powr } (-r :: \text{complex})))$
by $(\text{subst of_real_nat_power}) \text{ auto}$
also have $\dots = (\sum n. \text{Re } (n_+ \text{ powr } - (r :: \text{complex})))$ by *auto*
also have $\dots = \text{Re } (\sum n. n_+ \text{ powr } - (r :: \text{complex}))$
by $(\text{intro Re_suminf } [\text{symmetric}] \text{ summable_zeta})$
 $(\text{use assms in auto})$
also have $\dots = \text{Re } (\text{zeta } r)$
using *Re_complex_of_real zeta_conv_suminf assms* by *presburger*
finally show $?thesis$.
qed

lemma *inverse_zeta_bound'*:
assumes $1 < \text{Re } s$
shows $\|\text{inverse } (\text{zeta } s)\| \leq \text{Re } (\text{zeta } (\text{Re } s))$
proof –
write *moebius_mu* $(\langle \mu \rangle)$

let $?f = \lambda n :: \text{nat. } \mu (n_+) / (n_+) \text{ powr } s$
let $?g = \lambda n :: \text{nat. } (n_+) \text{ powr } - \text{Re } s$
have $\|\mu n :: \text{complex}\| \leq 1$ **for** n **by** $(\text{auto simp add: power_neg_one_If moebius_mu_def})$
hence $1: \|\?f n\| \leq ?g n$ **for** n
by $(\text{auto simp add: powr_minus norm_divide norm_powr_real_powr field_simps})$
have $\text{inverse } (zeta s) = (\sum n. ?f n)$
by $(\text{intro sums_unique inverse_zeta_sums assms})$
hence $\|\text{inverse } (zeta s)\| = \|\sum n. ?f n\|$ **by** auto
also have $\dots \leq (\sum n. ?g n)$ **by** $(\text{intro suminf_norm_bound summable_zeta_real assms } 1)$
finally show $?thesis$ **using** $\text{sums_zeta_real assms}$ **by** auto
qed

lemma $\text{zeta_bound}'$:

assumes $1 < \text{Re } s$
shows $\|\text{zeta } s\| \leq \text{Re } (zeta (\text{Re } s))$
proof –
let $?f = \lambda n :: \text{nat. } (n_+) \text{ powr } - s$
let $?g = \lambda n :: \text{nat. } (n_+) \text{ powr } - \text{Re } s$
have $\text{zeta } s = (\sum n. ?f n)$ **by** $(\text{intro sums_unique sums_zeta assms})$
hence $\|\text{zeta } s\| = \|\sum n. ?f n\|$ **by** auto
also have $\dots \leq (\sum n. ?g n)$
by $(\text{intro suminf_norm_bound summable_zeta_real assms})$
 $(\text{subst norm_nat_power, auto})$
also have $\dots = \text{Re } (zeta (\text{Re } s))$ **by** $(\text{subst sums_zeta_real})$ $(\text{use assms in auto})$
finally show $?thesis$.
qed

lemma $\text{zeta_bound_trivial}'$:

assumes $1 / 2 \leq \text{Re } s \wedge \text{Re } s \leq 2$
and $|Im s| \geq 1 / 11$
shows $\|\text{zeta } s\| \leq 12 + 2 * |Im s|$
proof –
have $\|\text{pre_zeta } 1 s\| \leq \|s\| / \text{Re } s$
by $(\text{rule pre_zeta_1_bound})$ $(\text{use assms in auto})$
also have $\dots \leq (|\text{Re } s| + |Im s|) / \text{Re } s$
proof –
have $\|s\| \leq |\text{Re } s| + |Im s|$ **using** cmod_le **by** auto
thus $?thesis$ **using** assms **by** $(\text{auto intro: divide_right_mono})$
qed
also have $\dots = 1 + |Im s| / \text{Re } s$
using assms **by** $(\text{simp add: field_simps})$
also have $\dots \leq 1 + |Im s| / (1 / 2)$
using assms **by** $(\text{intro add_left_mono divide_left_mono})$ auto
finally have $1: \|\text{pre_zeta } 1 s\| \leq 1 + 2 * |Im s|$ **by** auto
have $\|1 / (s - 1)\| = 1 / \|s - 1\|$ **by** $(\text{subst norm_divide})$ auto
also have $\dots \leq 11$ **proof** –
have $1 / 11 \leq |Im s|$ **by** $(\text{rule assms}(2))$
also have $\dots = |Im (s - 1)|$ **by** auto
also have $\dots \leq \|s - 1\|$ **by** $(\text{rule abs_Im_le_cmod})$
finally show $?thesis$ **by** $(\text{intro mult_imp_div_pos_le})$ auto
qed
finally have $2: \|1 / (s - 1)\| \leq 11$ **by** auto
have $\text{zeta } s = \text{pre_zeta } 1 s + 1 / (s - 1)$ **by** $(\text{intro zeta_pole_eq})$ $(\text{use assms in auto})$
moreover have $\|\dots\| \leq \|\text{pre_zeta } 1 s\| + \|1 / (s - 1)\|$ **by** $(\text{rule norm_triangle_ineq})$
ultimately have $\|\text{zeta } s\| \leq \dots$ **by** auto

also have $\dots \leq 12 + 2 * |Im s|$ **using** 1 2 **by** *auto*
 finally show *?thesis* .

qed

lemma *zeta_bound_gt_1*:

assumes $1 < Re s$

shows $\|zeta s\| \leq Re s / (Re s - 1)$

proof –

have $\|zeta s\| \leq Re (zeta (Re s))$ **by** (*intro zeta_bound' assms*)

also have $\dots \leq \|zeta (Re s)\|$ **by** (*rule complex_Re_le_cmod*)

also have $\dots = \|pre_zeta 1 (Re s) + 1 / (Re s - 1)\|$

by (*subst zeta_pole_eq*) (*use assms in auto*)

also have $\dots \leq \|pre_zeta 1 (Re s)\| + \|1 / (Re s - 1) :: complex\|$

by (*rule norm_triangle_ineq*)

also have $\dots \leq 1 + 1 / (Re s - 1)$

proof –

have $\|pre_zeta 1 (Re s)\| \leq \|Re s :: complex\| / Re (Re s)$

by (*rule pre_zeta_1_bound*) (*use assms in auto*)

also have $\dots = 1$ **using** *assms* **by** *auto*

moreover have $\|1 / (Re s - 1) :: complex\| = 1 / (Re s - 1)$

by (*subst norm_of_real*) (*use assms in auto*)

ultimately show *?thesis* **by** *auto*

qed

also have $\dots = Re s / (Re s - 1)$

using *assms* **by** (*auto simp add: field_simps*)

finally show *?thesis* .

qed

lemma *zeta_bound_trivial*:

assumes $1 / 2 \leq Re s$ **and** $|Im s| \geq 1 / 11$

shows $\|zeta s\| \leq 12 + 2 * |Im s|$

proof (*cases Re s ≤ 2*)

assume $Re s \leq 2$

thus *?thesis* **by** (*intro zeta_bound_trivial'*) (*use assms in auto*)

next

assume $\neg Re s \leq 2$

hence $*: Re s > 1$ $Re s > 2$ **by** *auto*

hence $\|zeta s\| \leq Re s / (Re s - 1)$ **by** (*intro zeta_bound_gt_1*)

also have $\dots \leq 2$ **using** $*$ **by** (*auto simp add: field_simps*)

also have $\dots \leq 12 + 2 * |Im s|$ **by** *auto*

finally show *?thesis* .

qed

lemma *zeta_nonzero_small_imag'*:

assumes $|Im s| \leq 13 / 22$ **and** $Re s \geq 1 / 2$ **and** $Re s < 1$

shows $zeta s \neq 0$

proof –

have $\|pre_zeta 1 s\| \leq (1 + \|s\| / Re s) / 2 * 1^{powr - Re s}$

by (*rule pre_zeta_bound*) (*use assms(2) in auto*)

also have $\dots \leq 129 / 100$ **proof** –

have $\|s\| / Re s \leq 79 / 50$

proof (*rule ccontr*)

assume $\neg \|s\| / Re s \leq 79 / 50$

hence $\sqrt{6241 / 2500} < \|s\| / Re s$ **by** (*simp add: real_sqrt_divide*)

also have $\dots = \|s\| / \sqrt{(Re s)^2}$ **using** *assms(2)* **by** *simp*

also have ... = $\text{sqrt}(1 + (\text{Im } s / \text{Re } s)^2)$

unfolding *cmod_def* using *assms(2)*

by (*auto simp add: real_sqrt_divide [symmetric] field_simps*
simp del: real_sqrt_abs)

finally have 1: $6241 / 2500 < 1 + (\text{Im } s / \text{Re } s)^2$ by *auto*

have $|\text{Im } s / \text{Re } s| \leq |6 / 5|$ using *assms* by (*auto simp add: field_simps abs_le_square_iff*)

hence $(\text{Im } s / \text{Re } s)^2 \leq (6 / 5)^2$ by (*subst (asm) abs_le_square_iff*)

hence 2: $1 + (\text{Im } s / \text{Re } s)^2 \leq 61 / 25$ unfolding *power2_eq_square* by *auto*

from 1 2 show *False* by *auto*

qed

hence $(1 + \|s\| / \text{Re } s) / 2 \leq (129 / 50) / 2$ by (*subst divide_right_mono*) *auto*

also have ... = $129 / 100$ by *auto*

finally show *?thesis* by *auto*

qed

finally have 1: $\|\text{pre_zeta } 1 \ s\| \leq 129 / 100$.

have $\|s - 1\| < 100 / 129$ proof -

from *assms* have $(\text{Re } (s - 1))^2 \leq (1 / 2)^2$ by (*simp add: abs_le_square_iff [symmetric]*)

moreover have $(\text{Im } (s - 1))^2 \leq (13 / 22)^2$ using *assms(1)* by (*simp add: abs_le_square_iff [symmetric]*)

ultimately have $(\text{Re } (s - 1))^2 + (\text{Im } (s - 1))^2 \leq 145 / 242$ by (*auto simp add: power2_eq_square*)

hence $\text{sqrt}((\text{Re } (s - 1))^2 + (\text{Im } (s - 1))^2) \leq \text{sqrt}(145 / 242)$ by (*rule real_sqrt_le_mono*)

also have ... < $\text{sqrt}((100 / 129)^2)$ by (*subst real_sqrt_less_iff*) (*simp add: power2_eq_square*)

finally show *?thesis* unfolding *cmod_def* by *auto*

qed

moreover have $\|s - 1\| \neq 0$ using *assms(3)* by *auto*

ultimately have 2: $\|1 / (s - 1)\| > 129 / 100$ by (*auto simp add: field_simps norm_divide*)

from 1 2 have $0 < \|1 / (s - 1)\| - \|\text{pre_zeta } 1 \ s\|$ by *auto*

also have ... $\leq \|\text{pre_zeta } 1 \ s + 1 / (s - 1)\|$ by (*subst add.commute*) (*rule norm_diff_ineq*)

also from *assms(3)* have $s \neq 1$ by *auto*

hence $\|\text{pre_zeta } 1 \ s + 1 / (s - 1)\| = \|\text{zeta } s\|$ using *zeta_pole_eq* by *auto*

finally show *?thesis* by *auto*

qed

lemma *zeta_nonzero_small_imag*:

assumes $|\text{Im } s| \leq 13 / 22$ and $\text{Re } s > 0$ and $s \neq 1$

shows $\text{zeta } s \neq 0$

proof -

consider $\text{Re } s \leq 1 / 2 \mid 1 / 2 \leq \text{Re } s \wedge \text{Re } s < 1 \mid \text{Re } s \geq 1$ by *fastforce*

thus *?thesis* proof cases

case 1 hence $\text{zeta } (1 - s) \neq 0$ using *assms* by (*intro zeta_nonzero_small_imag'*) *auto*

moreover case 1

ultimately show *?thesis* using *assms(2)* *zeta_zero_reflect_iff* by *auto*

next

case 2 thus *?thesis* using *assms(1)* by (*intro zeta_nonzero_small_imag'*) *auto*

next

case 3 thus *?thesis* using *zeta_Re_ge_1_nonzero* *assms(3)* by *auto*

qed

qed

lemma *inverse_zeta_bound*:

assumes $1 < \text{Re } s$

shows $\|\text{inverse } (\text{zeta } s)\| \leq \text{Re } s / (\text{Re } s - 1)$

proof -

have $\|\text{inverse } (\text{zeta } s)\| \leq \text{Re } (\text{zeta } (\text{Re } s))$ by (*intro inverse_zeta_bound' assms*)

also have ... $\leq \|\text{zeta } (\text{Re } s)\|$ by (*rule complex_Re_le_cmod*)

also have $\dots \leq \text{Re} (\text{Re } s) / (\text{Re} (\text{Re } s) - 1)$
 by $(\text{intro } \text{zeta_bound_gt_1}) (\text{use } \text{assms } \text{in } \text{auto})$
 also have $\dots = \text{Re } s / (\text{Re } s - 1)$ by auto
 finally show $?thesis$.
 qed

lemma deriv_zeta_bound :

fixes $s :: \text{complex}$
 assumes $Hr: 0 < r$ and $Hs: s \neq 1$
 and $hB: \bigwedge w. \|s - w\| = r \implies \|\text{pre_zeta } 1 \ w\| \leq B$
 shows $\|\text{deriv } \text{zeta } s\| \leq B / r + 1 / \|s - 1\|^2$
 proof -
 have $\|\text{deriv } \text{zeta } s\| = \|\text{deriv } (\text{pre_zeta } 1) \ s - 1 / (s - 1)^2\|$
 proof -
 let $?A = \text{UNIV} - \{1 :: \text{complex}\}$
 let $?f = \lambda s. \text{pre_zeta } 1 \ s + 1 / (s - 1)$
 let $?v = \text{deriv } (\text{pre_zeta } 1) \ s + (0 * (s - 1) - 1 * (1 - 0)) / (s - 1)^2$
 let $?v' = \text{deriv } (\text{pre_zeta } 1) \ s - 1 / (s - 1 :: \text{complex})^2$
 have $\forall z \in ?A. \text{zeta } z = \text{pre_zeta } 1 \ z + 1 / (z - 1)$
 by $(\text{auto } \text{intro}: \text{zeta_pole_eq})$
 hence $\forall_F z \text{ in } \text{nhds } s. \text{zeta } z = \text{pre_zeta } 1 \ z + 1 / (z - 1)$
 using Hs by $(\text{subst } \text{eventually_nhds}, \text{intro } \text{exI } [\text{where } x = ?A]) \text{ auto}$
 hence $\text{DERIV } \text{zeta } s :> ?v' = \text{DERIV } ?f \ s :> ?v'$
 by $(\text{intro } \text{DERIV_cong_ev}) \text{ auto}$
 moreover have $\text{DERIV } ?f \ s :> ?v$
 unfolding power2_eq_square
 by $(\text{intro } \text{derivative_intros } \text{field_differentiable_derivI } \text{holomorphic_pre_zeta}$
 $\text{holomorphic_on_imp_differentiable_at } [\text{where } s = ?A])$
 $(\text{use } Hs \text{ in } \text{auto})$
 moreover have $?v = ?v'$ by $(\text{auto } \text{simp } \text{add}: \text{field_simps})$
 ultimately have $\text{DERIV } \text{zeta } s :> ?v'$ by auto
 moreover have $\text{DERIV } \text{zeta } s :> \text{deriv } \text{zeta } s$
 by $(\text{intro } \text{field_differentiable_derivI } \text{field_differentiable_at_zeta})$
 $(\text{use } Hs \text{ in } \text{auto})$
 ultimately have $?v' = \text{deriv } \text{zeta } s$ by $(\text{rule } \text{DERIV_unique})$
 thus $?thesis$ by auto

qed

also have $\dots \leq \|\text{deriv } (\text{pre_zeta } 1) \ s\| + \|1 / (s - 1)^2\|$ by $(\text{rule } \text{norm_triangle_ineq4})$

also have $\dots \leq B / r + 1 / \|s - 1\|^2$

proof -

have $\|(\text{deriv } \sim 1) (\text{pre_zeta } 1) \ s\| \leq \text{fact } 1 * B / r \wedge 1$

by $(\text{intro } \text{Cauchy_inequality } \text{holomorphic_pre_zeta } \text{continuous_on_pre_zeta } \text{assms}) \text{ auto}$

thus $?thesis$ by $(\text{auto } \text{simp } \text{add}: \text{norm_divide } \text{norm_power})$

qed

finally show $?thesis$.

qed

lemma zeta_lower_bound :

assumes $0 < \text{Re } s$ and $s \neq 1$

shows $1 / \|s - 1\| - \|s\| / \text{Re } s \leq \|\text{zeta } s\|$

proof -

have $\|\text{pre_zeta } 1 \ s\| \leq \|s\| / \text{Re } s$ by $(\text{intro } \text{pre_zeta_1_bound } \text{assms})$

hence $1 / \|s - 1\| - \|s\| / \text{Re } s \leq \|1 / (s - 1)\| - \|\text{pre_zeta } 1 \ s\|$

using assms by $(\text{auto } \text{simp } \text{add}: \text{norm_divide})$

also have $\dots \leq \|\text{pre_zeta } 1 \ s + 1 / (s - 1)\|$

by (subst add.commute) (rule norm_diff_ineq)
also have ... = $\|\zeta s\|$ using *assms* by (subst zeta_pole_eq) auto
finally show *?thesis* .
qed

lemma logderiv_zeta_bound:

fixes $\sigma :: \text{real}$
assumes $1 < \sigma$ $\sigma \leq 23 / 20$
shows $\|\logderiv\ zeta\ \sigma\| \leq 5 / 4 * (1 / (\sigma - 1))$
proof -
have $\|\text{pre_zeta}\ 1\ s\| \leq \text{sqrt}\ 2$ if *: $\|\sigma - s\| = 1 / \text{sqrt}\ 2$ for $s :: \text{complex}$
proof -

have 1: $0 < \text{Re}\ s$ proof -
have $1 - \text{Re}\ s \leq \text{Re}\ (\sigma - s)$ using *assms*(1) by auto
also have $\text{Re}\ (\sigma - s) \leq \|\sigma - s\|$ by (rule complex_Re_le_cmod)
also have ... = $1 / \text{sqrt}\ 2$ by (rule *)
finally have $1 - 1 / \text{sqrt}\ 2 \leq \text{Re}\ s$ by auto
moreover have $0 < 1 - 1 / \text{sqrt}\ 2$ by auto
ultimately show *?thesis* by *linarith*

qed

hence $\|\text{pre_zeta}\ 1\ s\| \leq \|s\| / \text{Re}\ s$ by (rule pre_zeta_1_bound)

also have ... $\leq \text{sqrt}\ 2$ proof -

define $x\ y$ where $x \equiv \text{Re}\ s$ and $y \equiv \text{Im}\ s$
have $\text{sqrt}\ ((\sigma - x)^2 + y^2) = 1 / \text{sqrt}\ 2$
using * unfolding *cmod_def* *x_def* *y_def* by auto
also have ... = $\text{sqrt}\ (1 / 2)$ by (auto simp add: field_simps real_sqrt_mult [symmetric])
finally have 2: $x^2 + y^2 - 2*\sigma*x + \sigma^2 = 1 / 2$ by (auto simp add: field_simps power2_eq_square)
have $y^2 \leq x^2$ proof (rule ccontr)
assume $\neg y^2 \leq x^2$
hence $x^2 < y^2$ by auto
with 2 have $2*x^2 - 2*\sigma*x + \sigma^2 < 1 / 2$ by auto
hence $2 * (x - \sigma / 2)^2 < (1 - \sigma^2) / 2$ by (auto simp add: field_simps power2_eq_square)
also have ... < 0 using $\langle 1 < \sigma \rangle$ by auto
finally show *False* by auto

qed

moreover have $x \neq 0$ unfolding *x_def* using 1 by auto

ultimately have $\text{sqrt}\ ((x^2 + y^2) / x^2) \leq \text{sqrt}\ 2$ by (auto simp add: field_simps)

with 1 show *?thesis* unfolding *cmod_def* *x_def* *y_def* by (auto simp add: real_sqrt_divide)

qed

finally show *?thesis* .

qed

hence $\|\text{deriv}\ zeta\ \sigma\| \leq \text{sqrt}\ 2 / (1 / \text{sqrt}\ 2) + 1 / \|(\sigma :: \text{complex}) - 1\|^2$

by (intro deriv_zeta_bound) (use *assms*(1) in auto)

also have ... $\leq 2 + 1 / (\sigma - 1)^2$

by (subst in_Reals_norm) (use *assms*(1) in auto)

also have ... = $(2 * \sigma^2 - 4 * \sigma + 3) / (\sigma - 1)^2$

proof -

have $\sigma * \sigma - 2 * \sigma + 1 = (\sigma - 1) * (\sigma - 1)$ by (auto simp add: field_simps)

also have ... $\neq 0$ using *assms*(1) by auto

finally show *?thesis* by (auto simp add: power2_eq_square field_simps)

qed

finally have 1: $\|\text{deriv}\ zeta\ \sigma\| \leq (2 * \sigma^2 - 4 * \sigma + 3) / (\sigma - 1)^2$.

have $(2 - \sigma) / (\sigma - 1) = 1 / \|(\sigma :: \text{complex}) - 1\| - \|\sigma :: \text{complex}\| / \text{Re}\ \sigma$

using *assms*(1) by (auto simp add: field_simps in_Reals_norm)

also have ... $\leq \|\zeta\ \sigma\|$ by (rule zeta_lower_bound) (use *assms*(1) in auto)

finally have 2: $(2 - \sigma) / (\sigma - 1) \leq \|\zeta \sigma\|$.
have $4 * (2 * \sigma^2 - 4 * \sigma + 3) - 5 * (2 - \sigma) = 8 * (\sigma - 11 / 16)^2 - 57 / 32$
by (*auto simp add: field_simps power2_eq_square*)
also have $\dots \leq 0$ **proof** -
have $0 \leq \sigma - 11 / 16$ **using** *assms(1)* **by** *auto*
moreover have $\sigma - 11 / 16 \leq 37 / 80$ **using** *assms(2)* **by** *auto*
ultimately have $(\sigma - 11 / 16)^2 \leq (37 / 80)^2$ **by** *auto*
thus ?thesis **by** (*auto simp add: power2_eq_square*)
qed
finally have $4 * (2 * \sigma^2 - 4 * \sigma + 3) - 5 * (2 - \sigma) \leq 0$.
moreover have $0 < 2 - \sigma$ **using** *assms(2)* **by** *auto*
ultimately have 3: $(2 * \sigma^2 - 4 * \sigma + 3) / (2 - \sigma) \leq 5 / 4$ **by** (*subst pos_divide_le_eq*) *auto*
moreover have $0 \leq 2 * \sigma^2 - 4 * \sigma + 3$ **proof** -
have $0 \leq 2 * (\sigma - 1)^2 + 1$ **by** *auto*
also have $\dots = 2 * \sigma^2 - 4 * \sigma + 3$ **by** (*auto simp add: field_simps power2_eq_square*)
finally show ?thesis .
qed
moreover have $0 < (2 - \sigma) / (\sigma - 1)$ **using** *assms* **by** *auto*
ultimately have $\|\logderiv \zeta \sigma\| \leq ((2 * \sigma^2 - 4 * \sigma + 3) / (\sigma - 1)^2) / ((2 - \sigma) / (\sigma - 1))$
unfolding *logderiv_def* **using** 1 2 **by** (*subst norm_divide*) (*rule frac_le, auto*)
also have $\dots = (2 * \sigma^2 - 4 * \sigma + 3) / (2 - \sigma) * (1 / (\sigma - 1))$
by (*simp add: power2_eq_square*)
also have $\dots \leq 5 / 4 * (1 / (\sigma - 1))$
using 3 **by** (*rule mult_right_mono*) (*use assms(1) in auto*)
finally show ?thesis .
qed
lemma *Re_logderiv_zeta_bound*:
fixes $\sigma :: real$
assumes $1 < \sigma \wedge \sigma \leq 23 / 20$
shows $Re (\logderiv \zeta \sigma) \geq -5 / 4 * (1 / (\sigma - 1))$
proof -
have $- Re (\logderiv \zeta \sigma) = Re (- \logderiv \zeta \sigma)$ **by** *auto*
also have $Re (- \logderiv \zeta \sigma) \leq \|- \logderiv \zeta \sigma\|$ **by** (*rule complex_Re_le_cmod*)
also have $\dots = \|\logderiv \zeta \sigma\|$ **by** *auto*
also have $\dots \leq 5 / 4 * (1 / (\sigma - 1))$ **by** (*intro logderiv_zeta_bound assms*)
finally show ?thesis **by** *auto*
qed
locale *zeta_bound_param* =
fixes $\vartheta \varphi :: real \Rightarrow real$
assumes *zeta_bn'*: $\bigwedge z. 1 - \vartheta (Im z) \leq Re z \implies Im z \geq 1 / 11 \implies \|\zeta z\| \leq exp (\varphi (Im z))$
and *var_pos*: $\bigwedge t. 0 < \vartheta t \wedge \vartheta t \leq 1 / 2$
and *var_pos*: $\bigwedge t. 1 \leq \varphi t$
and *inv_var*: $\bigwedge t. \varphi t / \vartheta t \leq 1 / 960 * exp (\varphi t)$
and *mod*: *antimono* ϑ **and** *mo* φ : *mono* φ
begin
definition *region* $\equiv \{z. 1 - \vartheta (Im z) \leq Re z \wedge Im z \geq 1 / 11\}$
lemma *zeta_bn*: $\bigwedge z. z \in region \implies \|\zeta z\| \leq exp (\varphi (Im z))$
using *zeta_bn'* **unfolding** *region_def* **by** *auto*
lemma *var_pos'*: $\bigwedge t. 0 < \vartheta t \wedge \vartheta t \leq 1$
using *var_pos* **by** (*smt (verit) exp_ge_add_one_self exp_half_le2*)
lemma *var_pos'*: $\bigwedge t. 0 < \varphi t$ **using** *var_pos* **by** (*smt (verit, ccfv_SIG)*)
end

```

locale zeta_bound_param_1 = zeta_bound_param +
  fixes  $\gamma :: \text{real}$ 
  assumes  $\gamma\_cnd: \gamma \geq 13 / 22$ 
begin
  definition  $r$  where  $r \equiv \vartheta (2 * \gamma + 1)$ 
end

```

```

locale zeta_bound_param_2 = zeta_bound_param_1 +
  fixes  $\sigma \delta :: \text{real}$ 
  assumes  $\sigma\_cnd: \sigma \geq 1 + \exp (-\varphi(2 * \gamma + 1))$ 
  and  $\delta\_cnd: \delta = \gamma \vee \delta = 2 * \gamma$ 
begin
  definition  $s$  where  $s \equiv \text{Complex } \sigma \delta$ 
end

```

```

context zeta_bound_param_2 begin
declare  $dist\_complex\_def$  [simp]  $norm\_minus\_commute$  [simp]
declare  $legacy\_Complex\_simps$  [simp]

```

lemma *cball_lm*:

```

assumes  $z \in cball\ s\ r$ 
shows  $r \leq 1 \ |Re\ z - \sigma| \leq r \ |Im\ z - \delta| \leq r$ 
   $1 / 11 \leq Im\ z \ Im\ z \leq 2 * \gamma + r$ 

```

proof –

```

have  $|Re\ (z - s)| \leq \|z - s\| \ |Im\ (z - s)| \leq \|z - s\|$ 
  by (rule abs_Re_le_cmod) (rule abs_Im_le_cmod)
moreover have  $\|z - s\| \leq r$  using assms by auto
ultimately show  $1: |Re\ z - \sigma| \leq r \ |Im\ z - \delta| \leq r$  unfolding  $s\_def$  by auto
moreover have  $\exists: r \leq 1 / 2$  unfolding  $r\_def$  using  $\vartheta\_pos$  by auto
ultimately have  $2: |Re\ z - \sigma| \leq 1 / 2 \ |Im\ z - \delta| \leq 1 / 2$  by auto
moreover have  $\delta \leq 2 * \gamma$  using  $\delta\_cnd \ \gamma\_cnd$  by auto
ultimately show  $Im\ z \leq 2 * \gamma + r$  using  $1$  by auto
have  $1 / 11 \leq \delta - 1 / 2$  using  $\delta\_cnd \ \gamma\_cnd$  by auto
also have  $\dots \leq Im\ z$  using  $2$  by (auto simp del: Num.le_divide_eq_numeral1)
finally show  $1 / 11 \leq Im\ z$  .
from  $\exists$  show  $r \leq 1$  by auto

```

qed

lemma *cball_in_region*:

```

shows  $cball\ s\ r \subseteq region$ 

```

proof

```

fix  $z :: \text{complex}$ 
assume  $hz: z \in cball\ s\ r$ 
note  $lm = cball\_lm$  [OF  $hz$ ]
hence  $1 - \vartheta (Im\ z) \leq 1 - \vartheta (2 * \gamma + \vartheta (2 * \gamma + 1))$ 
  unfolding  $r\_def$  using  $mod\vartheta\ lm$  by (auto intro: antimonod)
also have  $\dots \leq 1 + \exp (-\varphi (2 * \gamma + 1)) - \vartheta (2 * \gamma + 1)$ 

```

proof –

```

have  $2 * \gamma + \vartheta (2 * \gamma + 1) \leq 2 * \gamma + 1$ 
  unfolding  $r\_def$  using  $\vartheta\_pos'$  by auto
hence  $\vartheta (2 * \gamma + 1) - \vartheta (2 * \gamma + \vartheta (2 * \gamma + 1)) \leq 0$ 
  using  $mod\vartheta$  by (auto intro: antimonod)
also have  $0 \leq \exp (-\varphi (2 * \gamma + 1))$  by auto
finally show ?thesis by auto

```

qed

also have $\dots \leq \sigma - r$ using σ_cnd unfolding r_def s_def by *auto*
also have $\dots \leq Re\ z$ using *lm* by *auto*
finally have $1 - \vartheta (Im\ z) \leq Re\ z$.
thus $z \in region$ unfolding $region_def$ using *lm* by *auto*
qed

lemma *Re_s_gt_1*:
shows $1 < Re\ s$
proof –
have *: $exp\ (-\ \varphi\ (2 * \gamma + 1)) > 0$ by *auto*
show *?thesis* using σ_cnd s_def by *auto* (use * in *linarith*)
qed

lemma *zeta_analytic_on_region*:
shows *zeta analytic_on region*
by (rule *analytic_zeta*) (unfold $region_def$, *auto*)

lemma *zeta_div_bound*:
assumes $z \in cball\ s\ r$
shows $\|zeta\ z / zeta\ s\| \leq exp\ (3 * \varphi\ (2 * \gamma + 1))$
proof –
let $?\varphi = \varphi\ (2 * \gamma + 1)$
have $\|zeta\ z\| \leq exp\ (\varphi\ (Im\ z))$ using *cball_in_region* *zeta_bn assms* by *auto*
also have $\dots \leq exp\ (?\varphi)$
proof –
have $Im\ z \leq 2 * \gamma + 1$ using *cball_lm* [*OF assms*] by *auto*
thus *?thesis* by *auto* (rule *monoD* [*OF mo\varphi*])
qed
also have $\|inverse\ (zeta\ s)\| \leq exp\ (2 * ?\varphi)$
proof –
have $\|inverse\ (zeta\ s)\| \leq Re\ s / (Re\ s - 1)$
by (intro *inverse_zeta_bound* *Re_s_gt_1*)
also have $\dots = 1 + 1 / (Re\ s - 1)$
using *Re_s_gt_1* by (auto simp add: *field_simps*)
also have $\dots \leq 1 + exp\ (?\varphi)$
proof –
have $Re\ s - 1 \geq exp\ (-?\varphi)$ using s_def σ_cnd by *auto*
hence $1 / (Re\ s - 1) \leq 1 / exp\ (-?\varphi)$
using *Re_s_gt_1* by (auto intro: *divide_left_mono*)
thus *?thesis* by (auto simp add: *exp_minus field_simps*)
qed
also have $\dots \leq exp\ (2 * ?\varphi)$ by (intro *exp_lemma_1 less_imp_le* φ_pos)
finally show *?thesis* .
qed
ultimately have $\|zeta\ z * inverse\ (zeta\ s)\| \leq exp\ (?\varphi) * exp\ (2 * ?\varphi)$
by (*subst norm_mult*, intro *mult_mono'*) *auto*
also have $\dots = exp\ (3 * ?\varphi)$ by (*subst exp_add* [*symmetric*]) *auto*
finally show *?thesis* by (auto simp add: *divide_inverse*)
qed

lemma *logderiv_zeta_bound*:
shows $Re\ (logderiv\ zeta\ s) \geq -24 * \varphi\ (2 * \gamma + 1) / r$
and $\bigwedge \beta. \sigma - r / 2 \leq \beta \implies zeta\ (Complex\ \beta\ \delta) = 0 \implies$
 $Re\ (logderiv\ zeta\ s) \geq -24 * \varphi\ (2 * \gamma + 1) / r + 1 / (\sigma - \beta)$
proof –

```

have 1: 0 < r unfolding r_def using  $\vartheta\_pos'$  by auto
have 2: 0 ≤ 3 *  $\varphi$  (2 *  $\gamma$  + 1) using  $\varphi\_pos'$  by (auto simp add: less_imp_le)
have 3: zeta s ≠ 0  $\wedge$  z. Re s < Re z  $\implies$  zeta z ≠ 0
  using Re_s_gt_1 by (auto intro!: zeta_Re_gt_1_nonzero)
have 4: zeta analytic_on cball s r
  by (rule analytic_on_subset;
      rule cball_in_region zeta_analytic_on_region)
have 5: z ∈ cball s r  $\implies$  ||zeta z / zeta s|| ≤ exp (3 *  $\varphi$  (2 *  $\gamma$  + 1))
  for z by (rule zeta_div_bound)
have 6: {} ⊆ {z ∈ cball s (r / 2). zeta z = 0  $\wedge$  Re z ≤ Re s} by auto
have 7: {Complex  $\beta$   $\delta$ } ⊆ {z ∈ cball s (r / 2). zeta z = 0  $\wedge$  Re z ≤ Re s}
  if  $\sigma - r / 2 \leq \beta$  zeta (Complex  $\beta$   $\delta$ ) = 0 for  $\beta$ 
proof -
  have  $\beta \leq \sigma$ 
    using zeta_Re_gt_1_nonzero [of Complex  $\beta$   $\delta$ ] Re_s_gt_1 that(2)
    unfolding s_def by fastforce
  thus ?thesis using that unfolding s_def by auto
qed
have - Re (logderiv zeta s) ≤ 8 * (3 *  $\varphi$  (2 *  $\gamma$  + 1)) / r + Re ( $\sum z \in \{\}$ . 1 / (z - s))
  by (intro lemma_3_9_beta3 1 2 3 4 5 6)
thus Re (logderiv zeta s) ≥ - 24 *  $\varphi$  (2 *  $\gamma$  + 1) / r by auto
show Re (logderiv zeta s) ≥ - 24 *  $\varphi$  (2 *  $\gamma$  + 1) / r + 1 / ( $\sigma - \beta$ )
  if *:  $\sigma - r / 2 \leq \beta$  zeta (Complex  $\beta$   $\delta$ ) = 0 for  $\beta$ 
proof -
  have bs:  $\beta \neq \sigma$  using *(2) 3(1) unfolding s_def by auto
  hence bs': 1 / ( $\beta - \sigma$ ) = - 1 / ( $\sigma - \beta$ ) by (auto simp add: field_simps)
  have inv_r: 1 / (Complex r 0) = Complex (1 / r) 0 if r ≠ 0 for r
    using that by (auto simp add: field_simps)
  have - Re (logderiv zeta s) ≤ 8 * (3 *  $\varphi$  (2 *  $\gamma$  + 1)) / r + Re ( $\sum z \in \{\text{Complex } \beta \delta\}$ . 1 / (z - s))
    by (intro lemma_3_9_beta3 1 2 3 4 5 7 *)
  thus ?thesis unfolding s_def
    by (auto simp add: field_simps)
    (subst (asm) inv_r, use bs bs' in auto)
qed
qed
end

```

context zeta_bound_param_1 begin

lemma zeta_nonzero_region':

```

assumes 1 + 1 / 960 * (r /  $\varphi$  (2 *  $\gamma$  + 1)) - r / 2 ≤  $\beta$ 
  and zeta (Complex  $\beta$   $\gamma$ ) = 0
shows 1 -  $\beta \geq 1 / 29760 * (r / \varphi (2 * \gamma + 1))$ 

```

proof -

```

let ? $\varphi$  =  $\varphi$  (2 *  $\gamma$  + 1) and ? $\vartheta$  =  $\vartheta$  (2 *  $\gamma$  + 1)
define  $\sigma$  where  $\sigma \equiv 1 + 1 / 960 * (r / \varphi (2 * \gamma + 1))$ 
define a where a  $\equiv - 5 / 4 * (1 / (\sigma - 1))$ 
define b where b  $\equiv - 24 * \varphi (2 * \gamma + 1) / r + 1 / (\sigma - \beta)$ 
define c where c  $\equiv - 24 * \varphi (2 * \gamma + 1) / r$ 
have 1 + exp (- ? $\varphi$ ) ≤  $\sigma$ 

```

proof -

```

have 960 * exp (- ? $\varphi$ ) = 1 / (1 / 960 * exp ? $\varphi$ )
  by (auto simp add: exp_add [symmetric] field_simps)

```

also have ... ≤ 1 / (? φ / ? ϑ) proof -

```

have ? $\varphi$  / ? $\vartheta$  ≤ 1 / 960 * exp ? $\varphi$  by (rule inv_ $\vartheta$ )
thus ?thesis by (intro divide_left_mono) (use  $\vartheta\_pos$   $\varphi\_pos'$  in auto)

```


qed
also have $\dots = r / \varphi$ **unfolding** r_def **by** *auto*
finally show *?thesis* **unfolding** σ_def **by** *auto*
qed
note $*$ = *this* γ_cnd
interpret z : $zeta_bound_param_2 \vartheta \varphi \gamma \sigma \gamma$ **by** (*standard, use * in auto*)
interpret z' : $zeta_bound_param_2 \vartheta \varphi \gamma \sigma 2 * \gamma$ **by** (*standard, use * in auto*)
have $r \leq 1$ **unfolding** r_def **using** ϑ_pos' [*of* $2 * \gamma + 1$] **by** *auto*
moreover have $1 \leq \varphi (2 * \gamma + 1)$ **using** φ_pos **by** *auto*
ultimately have $r / \varphi (2 * \gamma + 1) \leq 1$ **by** *auto*
moreover have $0 < r$ $0 < \varphi (2 * \gamma + 1)$ **unfolding** r_def **using** ϑ_pos' φ_pos' **by** *auto*
hence $0 < r / \varphi (2 * \gamma + 1)$ **by** *auto*
ultimately have $1: 1 < \sigma \leq 23 / 20$ **unfolding** σ_def **by** *auto*
hence $Re (\logderiv zeta \sigma) \geq a$ **unfolding** a_def **by** (*intro Re_logderiv_zeta_bound*)
hence $Re (\logderiv zeta (Complex \sigma 0)) \geq a$ **by** *auto*
moreover have $Re (\logderiv zeta z.s) \geq b$ **unfolding** b_def
by (*rule z.logderiv_zeta_bound*) (*use assms r_def sigma_def in auto*)
hence $Re (\logderiv zeta (Complex \sigma \gamma)) \geq b$ **unfolding** $z.s_def$ **by** *auto*
moreover have $Re (\logderiv zeta z'.s) \geq c$ **unfolding** c_def **by** (*rule z'.logderiv_zeta_bound*)
hence $Re (\logderiv zeta (Complex \sigma (2 * \gamma))) \geq c$ **unfolding** $z'.s_def$ **by** *auto*
ultimately have $3 * a + 4 * b + c$
 $\leq 3 * Re (\logderiv zeta (Complex \sigma 0)) + 4 * Re (\logderiv zeta (Complex \sigma \gamma))$
 $+ Re (\logderiv zeta (Complex \sigma (2 * \gamma)))$ **by** *auto*
also have $\dots \leq 0$ **by** (*rule logderiv_zeta_ineq, rule 1*)
finally have $3 * a + 4 * b + c \leq 0$.
hence $4 / (\sigma - \beta) \leq 15 / 4 * (1 / (\sigma - 1)) + 120 * \varphi (2 * \gamma + 1) / r$
unfolding a_def b_def c_def **by** *auto*
also have $\dots = 3720 * \varphi (2 * \gamma + 1) / r$ **unfolding** σ_def **by** *auto*
finally have 2 : $inverse (\sigma - \beta) \leq 930 * \varphi (2 * \gamma + 1) / r$ **by** (*auto simp add: inverse_eq_divide*)
have 3 : $\sigma - \beta \geq 1 / 930 * (r / \varphi (2 * \gamma + 1))$
proof –
have $1 / 930 * (r / \varphi (2 * \gamma + 1)) = 1 / (930 * (\varphi (2 * \gamma + 1) / r))$
by (*auto simp add: field_simps*)
also have $\dots \leq \sigma - \beta$ **proof** –
have $\beta \leq 1$ **using** $assms(2)$ $zeta_Re_gt_1_nonzero$ [*of* $Complex \beta \gamma$] **by** *fastforce*
also have $1 < \sigma$ **by** (*rule 1*)
finally have $\beta < \sigma$.
thus *?thesis* **using** 2 **by** (*auto intro: inverse_le_imp_le*)
qed
finally show *?thesis* .
qed
show *?thesis* **proof** –
let $?x = r / \varphi (2 * \gamma + 1)$
have $1 / 29760 * ?x = 1 / 930 * ?x - 1 / 960 * ?x$ **by** *auto*
also have $\dots \leq (\sigma - \beta) - (\sigma - 1)$ **using** 3 **by** (*subst (2) sigma_def*) *auto*
also have $\dots = 1 - \beta$ **by** *auto*
finally show *?thesis* .
qed
qed
lemma $zeta_nonzero_region$:
assumes $zeta (Complex \beta \gamma) = 0$
shows $1 - \beta \geq 1 / 29760 * (r / \varphi (2 * \gamma + 1))$
proof (*cases* $1 + 1 / 960 * (r / \varphi (2 * \gamma + 1)) - r / 2 \leq \beta$)
case *True*

```

thus ?thesis using assms by (rule zeta_nonzero_region)
next
case False
let ?x = r /  $\varphi$  (2 *  $\gamma$  + 1)
assume 1:  $\neg 1 + 1 / 960 * ?x - r / 2 \leq \beta$ 
have 0 < r using  $\vartheta\_pos'$  unfolding r_def by auto
hence 1 / 930 * ?x  $\leq r / 2$ 
  using  $\varphi\_pos$  [of 2 *  $\gamma$  + 1] by (auto intro!: mult_imp_div_pos_le)
hence 1 / 29760 * ?x  $\leq r / 2 - 1 / 960 * ?x$  by auto
also have ...  $\leq 1 - \beta$  using 1 by auto
finally show ?thesis .
qed
end

context zeta_bound_param begin
theorem zeta_nonzero_region:
  assumes zeta (Complex  $\beta$   $\gamma$ ) = 0 and Complex  $\beta$   $\gamma \neq 1$ 
  shows  $1 - \beta \geq 1 / 29760 * (\vartheta (2 * |\gamma| + 1) / \varphi (2 * |\gamma| + 1))$ 
proof (cases  $|\gamma| \geq 13 / 22$ )
case True
assume 1:  $13 / 22 \leq |\gamma|$ 
have 2: zeta (Complex  $\beta$   $|\gamma|$ ) = 0
proof (cases  $\gamma \geq 0$ )
case True thus ?thesis using assms by auto
next
case False thus ?thesis by (auto simp add: complex_cnj [symmetric] intro: assms)
qed
interpret z: zeta_bound_param_1  $\vartheta$   $\varphi$   $\langle |\gamma| \rangle$  by standard (use 1 in auto)
show ?thesis by (intro z.zeta_nonzero_region [unfolded z.r_def] 2)
next
case False
hence 1:  $|\gamma| \leq 13 / 22$  by auto
show ?thesis
proof (cases  $0 < \beta$ , rule ccontr)
case True thus False using zeta_nonzero_small_imag [of Complex  $\beta$   $\gamma$ ] assms 1 by auto
next
have 0 <  $\vartheta (2 * |\gamma| + 1) \vartheta (2 * |\gamma| + 1) \leq 1$   $1 \leq \varphi (2 * |\gamma| + 1)$ 
  using  $\vartheta\_pos'$   $\varphi\_pos$  by auto
hence 1 / 29760 * ( $\vartheta (2 * |\gamma| + 1) / \varphi (2 * |\gamma| + 1)$ )  $\leq 1$  by auto
also case False hence  $1 \leq 1 - \beta$  by auto
finally show ?thesis .
qed
qed
end

lemma zeta_bound_param_nonneg:
  fixes  $\vartheta$   $\varphi$  :: real  $\Rightarrow$  real
  assumes zeta_bn!:  $\bigwedge z. 1 - \vartheta (\text{Im } z) \leq \text{Re } z \implies \text{Im } z \geq 1 / 11 \implies \|\text{zeta } z\| \leq \exp (\varphi (\text{Im } z))$ 
  and  $\vartheta\_pos$ :  $\bigwedge t. 0 \leq t \implies 0 < \vartheta t \wedge \vartheta t \leq 1 / 2$ 
  and  $\varphi\_pos$ :  $\bigwedge t. 0 \leq t \implies 1 \leq \varphi t$ 
  and inv  $\vartheta$ :  $\bigwedge t. 0 \leq t \implies \varphi t / \vartheta t \leq 1 / 960 * \exp (\varphi t)$ 
  and mod $\vartheta$ :  $\bigwedge x y. 0 \leq x \implies x \leq y \implies \vartheta y \leq \vartheta x$ 
  and mo $\varphi$ :  $\bigwedge x y. 0 \leq x \implies x \leq y \implies \varphi x \leq \varphi y$ 
  shows zeta_bound_param ( $\lambda t. \vartheta (\max 0 t)$ ) ( $\lambda t. \varphi (\max 0 t)$ )
  by standard (insert assms, auto simp add: antimono_def mono_def)

```

interpretation *classical_zeta_bound*:

zeta_bound_param $\lambda t. 1 / 2 \lambda t. 4 * \ln (12 + 2 * \max 0 t)$

proof –

define $\vartheta :: \text{real} \Rightarrow \text{real}$ **where** $\vartheta \equiv \lambda t. 1 / 2$

define $\varphi :: \text{real} \Rightarrow \text{real}$ **where** $\varphi \equiv \lambda t. 4 * \ln (12 + 2 * t)$

have *zeta_bound_param* ($\lambda t. \vartheta (\max 0 t)$) ($\lambda t. \varphi (\max 0 t)$)

proof (*rule zeta_bound_param_nonneg*)

fix z **assume** $*$: $1 - \vartheta (\text{Im } z) \leq \text{Re } z \text{ Im } z \geq 1 / 11$

have $\|zeta z\| \leq 12 + 2 * |\text{Im } z|$

using $*$ **unfolding** ϑ_def **by** (*intro zeta_bound_trivial*) *auto*

also have $\dots = \exp (\ln (12 + 2 * \text{Im } z))$ **using** $*(2)$ **by** *auto*

also have $\dots \leq \exp (\varphi (\text{Im } z))$ **proof** –

have $0 \leq \ln (12 + 2 * \text{Im } z)$ **using** $*(2)$ **by** *auto*

thus *?thesis* **unfolding** φ_def **by** *auto*

qed

finally show $\|zeta z\| \leq \exp (\varphi (\text{Im } z))$.

next

fix $t :: \text{real}$ **assume** $*$: $0 \leq t$

have $\varphi t / \vartheta t = 8 * \ln (12 + 2 * t)$ **unfolding** φ_def ϑ_def **by** *auto*

also have $\dots \leq 8 * (5 / 2 + t)$

proof –

have $\ln (12 + 2 * t) = \ln (12 * (1 + t / 6))$ **by** *auto*

also have $\dots = \ln 12 + \ln (1 + t / 6)$ **by** (*rule ln_mult*) (*use * in auto*)

also have $\dots \leq 5 / 2 + t / 6$

proof (*rule add_mono*)

have $(144 :: \text{real}) < (271 / 100) ^ 5$

by (*simp add: power_numeral_reduce*)

also have $271 / 100 < \exp (1 :: \text{real})$

using *e_approx_32* **by** (*simp add: abs_if_split: if_split_asm*)

hence $(271 / 100) ^ 5 < \exp (1 :: \text{real}) ^ 5$

by (*rule power_strict_mono*) *auto*

also have $\dots = \exp ((5 :: \text{nat}) * (1 :: \text{real}))$

by (*rule exp_of_nat_mult [symmetric]*)

also have $\dots = \exp (5 :: \text{real})$

by *auto*

finally have $\exp (\ln (12 :: \text{real}) * (2 :: \text{nat})) \leq \exp 5$

by (*subst exp_of_nat2_mult*) *auto*

thus $\ln (12 :: \text{real}) \leq 5 / 2$

by *auto*

show $\ln (1 + t / 6) \leq t / 6$

by (*intro ln_add_one_self_le_self*) (*use * in auto*)

qed

finally show *?thesis* **using** $*$ **by** *auto*

qed

also have $\dots \leq 1 / 960 * \exp (\varphi t)$

proof –

have $8 * (5 / 2 + t) - 1 / 960 * (12 + 2 * t) ^ 4$

$= -(1 / 60 * t ^ 4 + 2 / 5 * t ^ 3 + 18 / 5 * t ^ 2 + 32 / 5 * t + 8 / 5)$

by (*simp add: power_numeral_reduce field_simps*)

also have $\dots \leq 0$ **using** $*$

by (*subst neg_le_0_iff_le*) (*auto intro: add_nonneg_nonneg*)

moreover have $\exp (\varphi t) = (12 + 2 * t) ^ 4$

proof –

have $\exp (\varphi t) = (12 + 2 * t) \text{ powr } (\text{real } 4)$ **unfolding** φ_def *powr_def* **using** $*$ **by** *auto*

```

    also have ... = (12 + 2 * t) ^ 4 by (rule powr_realpow) (use * in auto)
    finally show ?thesis .
qed
ultimately show ?thesis by auto
qed
finally show  $\varphi t / \vartheta t \leq 1 / 960 * \exp(\varphi t)$  .
next
fix t :: real assume *: 0 ≤ t
have (1 :: real) ≤ 4 * 1 by auto
also have ... ≤ 4 * ln 12
proof -
  have  $\exp(1 :: real) \leq 3$  by (rule exp_le)
  also have ... ≤  $\exp(\ln 12)$  by auto
  finally have (1 :: real) ≤ ln 12 using exp_le_cancel_iff by blast
  thus ?thesis by auto
qed
also have ... ≤ 4 * ln (12 + 2 * t) using * by auto
finally show 1 ≤  $\varphi t$  unfolding  $\varphi\_def$  .
next
show  $\bigwedge t. 0 < \vartheta t \wedge \vartheta t \leq 1 / 2$ 
   $\bigwedge x y. 0 \leq x \implies x \leq y \implies \vartheta y \leq \vartheta x$ 
   $\bigwedge x y. 0 \leq x \implies x \leq y \implies \varphi x \leq \varphi y$ 
  unfolding  $\vartheta\_def$   $\varphi\_def$  by auto
qed
thus zeta_bound_param ( $\lambda t. 1 / 2$ ) ( $\lambda t. 4 * \ln(12 + 2 * \max 0 t)$ )
  unfolding  $\vartheta\_def$   $\varphi\_def$  by auto
qed

theorem zeta_nonzero_region:
  assumes zeta (Complex  $\beta \gamma$ ) = 0 and Complex  $\beta \gamma \neq 1$ 
  shows  $1 - \beta \geq C_1 / \ln(|\gamma| + 2)$ 
proof -
  have  $1 / 952320 * (1 / \ln(|\gamma| + 2))$ 
    ≤  $1 / 29760 * (1 / 2 / (4 * \ln(12 + 2 * \max 0 (2 * |\gamma| + 1))))$  (is ?x ≤ ?y)
  proof -
    have  $\ln(14 + 4 * |\gamma|) \leq 4 * \ln(|\gamma| + 2)$  by (rule ln_bound_1) auto
    hence  $1 / 238080 / (4 * \ln(|\gamma| + 2)) \leq 1 / 238080 / (\ln(14 + 4 * |\gamma|))$ 
      by (intro divide_left_mono) auto
    also have ... = ?y by auto
    finally show ?thesis by auto
  qed
  also have ... ≤  $1 - \beta$  by (intro classical_zeta_bound.zeta_nonzero_region assms)
  finally show ?thesis unfolding PNT_const_C1_def by auto
qed

unbundle no_pnt_notation
end
theory PNT_Subsummable
imports
  PNT_Remainder_Library
begin
unbundle pnt_notation

definition has_subsum where has_subsum f S x ≡ ( $\lambda n. \text{if } n \in S \text{ then } f n \text{ else } 0$ ) sums x
definition subsum where subsum f S ≡  $\sum n. \text{if } n \in S \text{ then } f n \text{ else } 0$ 

```

definition *subsummable* (**infix** *subsummable* 50)
where *f subsummable S* \equiv *summable* ($\lambda n. \text{if } n \in S \text{ then } f \ n \text{ else } 0$)

syntax *_subsum* :: *pttrn* \Rightarrow *nat set* \Rightarrow 'a \Rightarrow 'a
 $((2\sum _ \in (_) ./ _) [0, 0, 10] 10)$

translations

$\sum _ x \in S. t \Rightarrow$ *CONST subsum* ($\lambda x. t$) *S*

syntax *_subsum_prop* :: *pttrn* \Rightarrow *bool* \Rightarrow 'a \Rightarrow 'a
 $((2\sum _ | (_) ./ _) [0, 0, 10] 10)$

translations

$\sum _ x | P. t \Rightarrow$ *CONST subsum* ($\lambda x. t$) {*x. P*}

syntax *_subsum_ge* :: *pttrn* \Rightarrow *nat* \Rightarrow 'a \Rightarrow 'a
 $((2\sum _ \geq _ ./ _) [0, 0, 10] 10)$

translations

$\sum _ x \geq n. t \Rightarrow$ *CONST subsum* ($\lambda x. t$) {*n..*}

lemma *has_subsum_finite*:

finite F \Longrightarrow *has_subsum f F* (*sum f F*)

unfolding *has_subsum_def* **by** (*rule sums_If_finite_set*)

lemma *has_subsum_If_finite_set*:

assumes *finite F*

shows *has_subsum* ($\lambda n. \text{if } n \in F \text{ then } f \ n \text{ else } 0$) *A* (*sum f (F \cap A)*)

proof –

have *F \cap A* = {*x. x \in A \wedge x \in F*} **by** *auto*

thus *?thesis* **unfolding** *has_subsum_def* **using** *assms*

by (*auto simp add: if_if_eq_conj intro!: sums_If_finite*)

qed

lemma *has_subsum_If_finite*:

assumes *finite* {*n \in A. p n*}

shows *has_subsum* ($\lambda n. \text{if } p \ n \text{ then } f \ n \text{ else } 0$) *A* (*sum f {n \in A. p n}*)

unfolding *has_subsum_def* **using** *assms*

by (*auto simp add: if_if_eq_conj intro!: sums_If_finite*)

lemma *has_subsum_univ*:

f sums v \Longrightarrow *has_subsum f UNIV v*

unfolding *has_subsum_def* **by** *auto*

lemma *subsumI*:

fixes *f* :: *nat* \Rightarrow 'a :: {*t2_space, comm_monoid_add*}

shows *has_subsum f A x* \Longrightarrow *x = subsum f A*

unfolding *has_subsum_def subsum_def* **by** (*intro sums_unique*)

lemma *has_subsum_summable*:

has_subsum f A x \Longrightarrow *f subsummable A*

unfolding *has_subsum_def subsummable_def* **by** (*rule sums_summable*)

lemma *subsummable_sums*:

fixes *f* :: *nat* \Rightarrow 'a :: {*comm_monoid_add, t2_space*}

shows *f subsummable S* \Longrightarrow *has_subsum f S* (*subsum f S*)

unfolding *subsummable_def has_subsum_def subsum_def* **by** (*intro summable_sums*)

lemma *has_subsum_diff_finite*:

fixes $S :: 'a :: \{topological_ab_group_add, t2_space\}$
assumes *finite F has_subsum f A S F* $F \subseteq A$
shows *has_subsum f (A - F) (S - sum f F)*

proof –

define p **where** $p\ n \equiv$ *if* $n \in F$ *then* 0 *else* (*if* $n \in A$ *then* $f\ n$ *else* 0) **for** n
define q **where** $q\ n \equiv$ *if* $n \in A - F$ *then* $f\ n$ *else* 0 **for** n
have $F \cap A = F$ **using** *assms(3)* **by** *auto*
hence p *sums* $(S - sum\ f\ F)$
using *assms unfolding p_def has_subsum_def*
by (*auto intro: sums_If_finite_set' [where ?S = S]*
simp: sum_negf sum.inter_restrict [symmetric])
moreover **have** $p = q$ **unfolding** $p_def\ q_def$ **by** *auto*
finally **show** *?thesis unfolding q_def has_subsum_def* **by** *auto*

qed

lemma *subsum_split*:

fixes $f :: nat \Rightarrow 'a :: \{topological_ab_group_add, t2_space\}$
assumes *f subsummable A finite F F* $F \subseteq A$
shows *subsum f A = sum f F + subsum f (A - F)*

proof –

from *assms(1)* **have** *has_subsum f A (subsum f A)* **by** (*intro subsummable_sums*)
hence *has_subsum f (A - F) (subsum f A - sum f F)*
using *assms* **by** (*intro has_subsum_diff_finite*)
hence *subsum f A - sum f F = subsum f (A - F)* **by** (*rule subsumI*)
thus *?thesis* **by** (*auto simp add: algebra_simps*)

qed

lemma *has_subsum_zero* [*simp*]: *has_subsum* $(\lambda n. 0)$ $A\ 0$ **unfolding** *has_subsum_def* **by** *auto*

lemma *zero_subsummable* [*simp*]: $(\lambda n. 0)$ *subsummable A* **unfolding** *subsummable_def* **by** *auto*

lemma *zero_subsum* [*simp*]: $(\sum 'n \in A. 0 :: 'a :: \{comm_monoid_add, t2_space\}) = 0$ **unfolding** *subsum_def* **by** *auto*

lemma *has_subsum_minus*:

fixes $f :: nat \Rightarrow 'a :: real_normed_vector$
assumes *has_subsum f A a has_subsum g A b*
shows *has_subsum* $(\lambda n. f\ n - g\ n)$ $A\ (a - b)$

proof –

define p **where** $p\ n =$ (*if* $n \in A$ *then* $f\ n$ *else* 0) **for** n
define q **where** $q\ n =$ (*if* $n \in A$ *then* $g\ n$ *else* 0) **for** n
have $(\lambda n. p\ n - q\ n)$ *sums* $(a - b)$
using *assms unfolding p_def q_def has_subsum_def* **by** (*intro sums_diff*)
moreover **have** (*if* $n \in A$ *then* $f\ n - g\ n$ *else* 0) $= p\ n - q\ n$ **for** n
unfolding $p_def\ q_def$ **by** *auto*
ultimately **show** *?thesis unfolding has_subsum_def* **by** *auto*

qed

lemma *subsum_minus*:

assumes *f subsummable A g subsummable A*
shows *subsum f A - subsum g A =* $(\sum 'n \in A. f\ n - g\ n :: 'a :: real_normed_vector)$
by (*intro subsumI has_subsum_minus subsummable_sums assms*)

lemma *subsummable_minus*:

assumes *f subsummable A g subsummable A*
shows $(\lambda n. f\ n - g\ n :: 'a :: real_normed_vector)$ *subsummable A*

by (auto intro: has_subsum_summable has_subsum_minus subsummable_sums assms)

lemma *has_subsum_uminus:*

assumes *has_subsum* $f A a$

shows *has_subsum* $(\lambda n. - f n :: 'a :: \text{real_normed_vector}) A (- a)$

proof –

have *has_subsum* $(\lambda n. 0 - f n) A (0 - a)$

by (intro *has_subsum_minus*) (use *assms* in *auto*)

thus ?thesis by *auto*

qed

lemma *subsum_uminus:*

f subsummable $A \implies - \text{subsum } f A = (\sum 'n \in A. - f n :: 'a :: \text{real_normed_vector})$

by (intro *subsumI has_subsum_uminus subsummable_sums*)

lemma *subsummable_uminus:*

f subsummable $A \implies (\lambda n. - f n :: 'a :: \text{real_normed_vector})$ subsummable A

by (auto intro: *has_subsum_summable has_subsum_uminus subsummable_sums*)

lemma *has_subsum_add:*

fixes $f :: \text{nat} \Rightarrow 'a :: \text{real_normed_vector}$

assumes *has_subsum* $f A a$ *has_subsum* $g A b$

shows *has_subsum* $(\lambda n. f n + g n) A (a + b)$

proof –

have *has_subsum* $(\lambda n. f n - - g n) A (a - - b)$

by (intro *has_subsum_minus has_subsum_uminus assms*)

thus ?thesis by *auto*

qed

lemma *subsum_add:*

assumes f subsummable $A g$ subsummable A

shows $\text{subsum } f A + \text{subsum } g A = (\sum 'n \in A. f n + g n :: 'a :: \text{real_normed_vector})$

by (intro *subsumI has_subsum_add subsummable_sums assms*)

lemma *subsummable_add:*

assumes f subsummable $A g$ subsummable A

shows $(\lambda n. f n + g n :: 'a :: \text{real_normed_vector})$ subsummable A

by (auto intro: *has_subsum_summable has_subsum_add subsummable_sums assms*)

lemma *subsum_cong:*

$(\bigwedge x. x \in A \implies f x = g x) \implies \text{subsum } f A = \text{subsum } g A$

unfolding *subsum_def* by (intro *suminf_cong*) *auto*

lemma *subsummable_cong:*

fixes $f :: \text{nat} \Rightarrow 'a :: \text{real_normed_vector}$

shows $(\bigwedge x. x \in A \implies f x = g x) \implies (f \text{ subsummable } A) = (g \text{ subsummable } A)$

unfolding *subsummable_def* by (intro *summable_cong*) *auto*

lemma *subsum_norm_bound:*

fixes $f :: \text{nat} \Rightarrow 'a :: \text{banach}$

assumes g subsummable $A \bigwedge n. n \in A \implies \|f n\| \leq g n$

shows $\|\text{subsum } f A\| \leq \text{subsum } g A$

using *assms* unfolding *subsummable_def subsum_def*

by (intro *suminf_norm_bound*) *auto*

lemma *eval_fds_subsum*:
fixes $f :: 'a :: \{\text{nat_power}, \text{banach}, \text{real_normed_field}\}$ fds
assumes $fds_converges\ f\ s$
shows $has_subsum\ (\lambda n. fds_nth\ f\ n / nat_power\ n\ s)\ \{1..\}\ (eval_fds\ f\ s)$
proof –
let $?f = \lambda n. fds_nth\ f\ n / nat_power\ n\ s$
let $?v = eval_fds\ f\ s$
have $has_subsum\ (\lambda n. ?f\ n)\ UNIV\ ?v$
by (*intro has_subsum_univ fds_converges_iff [THEN iffD1] assms*)
hence $has_subsum\ ?f\ (UNIV - \{0\})\ (?v - sum\ ?f\ \{0\})$
by (*intro has_subsum_diff_finite*) *auto*
moreover **have** $UNIV - \{0 :: nat\} = \{1..\}$ **by** *auto*
ultimately **show** $?thesis$ **by** *auto*
qed

lemma *fds_abs_subsummable*:
fixes $f :: 'a :: \{\text{nat_power}, \text{banach}, \text{real_normed_field}\}$ fds
assumes $fds_abs_converges\ f\ s$
shows $(\lambda n. \|fds_nth\ f\ n / nat_power\ n\ s\|)\ subsummable\ \{1..\}$
proof –
have $summable\ (\lambda n. \|fds_nth\ f\ n / nat_power\ n\ s\|)$
by (*subst fds_abs_converges_def [symmetric]*) (*rule assms*)
moreover **have** $\|fds_nth\ f\ n / nat_power\ n\ s\| = 0$ **when** $\neg 1 \leq n$ **for** n
proof –
have $n = 0$ **using** *that* **by** *auto*
thus $?thesis$ **by** *auto*
qed
hence $(\lambda n. \text{if } 1 \leq n \text{ then } \|fds_nth\ f\ n / nat_power\ n\ s\| \text{ else } 0)$
 $= (\lambda n. \|fds_nth\ f\ n / nat_power\ n\ s\|)$ **by** *auto*
ultimately **show** $?thesis$ **unfolding** *subsummable_def* **by** *auto*
qed

lemma *subsum_mult2*:
fixes $f :: nat \Rightarrow 'a :: real_normed_algebra$
shows $f\ subsummable\ A \implies (\sum 'x \in A. f\ x * c) = subsum\ f\ A * c$
unfolding *subsum_def* *subsummable_def*
by (*subst suminf_mult2*) (*auto intro: suminf_cong*)

lemma *subsummable_mult2*:
fixes $f :: nat \Rightarrow 'a :: real_normed_algebra$
assumes $f\ subsummable\ A$
shows $(\lambda x. f\ x * c)\ subsummable\ A$
proof –
have $summable\ (\lambda n. (\text{if } n \in A \text{ then } f\ n \text{ else } 0) * c)$ (**is** $?P$)
using *assms* **unfolding** *subsummable_def* **by** (*intro summable_mult2*)
moreover **have** $?P = ?thesis$
unfolding *subsummable_def* **by** (*rule summable_cong*) *auto*
ultimately **show** $?thesis$ **by** *auto*
qed

lemma *subsum_ge_limit*:
 $lim\ (\lambda N. \sum n = m..N. f\ n) = (\sum 'n \geq m. f\ n)$
proof –
define g **where** $g\ n \equiv \text{if } n \in \{m..\} \text{ then } f\ n \text{ else } 0$ **for** n
have $(\sum n. g\ n) = lim\ (\lambda N. \sum n < N. g\ n)$ **by** (*rule suminf_eq_lim*)

also have $\dots = \lim (\lambda N. \sum_{n < N + 1}. g\ n)$
unfolding *lim_def* **using** *LIMSEQ_ignore_initial_segment LIMSEQ_offset*
by (*intro The_cong iffI*) *blast*
also have $\dots = \lim (\lambda N. \sum_{n = m..N}. f\ n)$
proof –
have $\{x. x < N + 1 \wedge m \leq x\} = \{m..N\}$ **for** *N* **by** *auto*
thus *?thesis* **unfolding** *g_def* **by** (*subst sum.inter_filter [symmetric]*) *auto*
qed
finally show *?thesis* **unfolding** *subsum_def g_def* **by** *auto*
qed

lemma *has_subsum_ge_limit*:
fixes *f :: nat => 'a :: {t2_space, comm_monoid_add, topological_space}*
assumes $((\lambda N. \sum_{n = m..N}. f\ n) \longrightarrow l)$ *at_top*
shows *has_subsum f {m..} l*
proof –
define *g* **where** $g\ n \equiv \text{if } n \in \{m..\} \text{ then } f\ n \text{ else } 0$ **for** *n*
have $((\lambda N. \sum_{n < N + 1}. g\ n) \longrightarrow l)$ *at_top*
proof –
have $\{x. x < N + 1 \wedge m \leq x\} = \{m..N\}$ **for** *N* **by** *auto*
with *assms* **show** *?thesis*
unfolding *g_def* **by** (*subst sum.inter_filter [symmetric]*) *auto*
qed
hence $((\lambda N. \sum_{n < N}. g\ n) \longrightarrow l)$ *at_top* **by** (*rule LIMSEQ_offset*)
thus *?thesis* **unfolding** *has_subsum_def sums_def g_def* **by** *auto*
qed

lemma *eval_fds_complex*:
fixes *f :: complex fds*
assumes *fds_converges f s*
shows *has_subsum* $(\lambda n. \text{fds_nth } f\ n / n \text{ nat_powr } s)$ $\{1..\}$ (*eval_fds f s*)
proof –
have *has_subsum* $(\lambda n. \text{fds_nth } f\ n / \text{nat_power } n\ s)$ $\{1..\}$ (*eval_fds f s*)
by (*intro eval_fds_subsum assms*)
thus *?thesis* **unfolding** *nat_power_complex_def* .
qed

lemma *eval_fds_complex_subsum*:
fixes *f :: complex fds*
assumes *fds_converges f s*
shows *eval_fds f s* = $(\sum 'n \geq 1. \text{fds_nth } f\ n / n \text{ nat_powr } s)$
 $(\lambda n. \text{fds_nth } f\ n / n \text{ nat_powr } s)$ *subsummable* $\{1..\}$
proof (*goal_cases*)
case 1 **show** *?case* **by** (*intro subsumI eval_fds_complex assms*)
case 2 **show** *?case* **by** (*intro has_subsum_summable*) (*rule eval_fds_complex assms*) +
qed

lemma *has_sum_imp_has_subsum*:
fixes *x :: 'a :: {comm_monoid_add, t2_space}*
assumes (*f has_sum x*) *A*
shows *has_subsum f A x*
proof –
have $(\forall_F x \text{ in } \text{at_top}. \text{sum } f\ (\{..<x\} \cap A) \in S)$
when *open S x \in S* **for** *S*
proof –

```

have  $\forall S. \text{open } S \longrightarrow x \in S \longrightarrow (\forall_F x \text{ in } \text{finite\_subsets\_at\_top } A. \text{sum } f x \in S)$ 
  using assms unfolding has_sum_def tendsto_def .
hence  $\forall_F x \text{ in } \text{finite\_subsets\_at\_top } A. \text{sum } f x \in S$  using that by auto
then obtain  $X$  where  $hX: \text{finite } X \ X \subseteq A$ 
  and  $hY: \bigwedge Y. \text{finite } Y \implies X \subseteq Y \implies Y \subseteq A \implies \text{sum } f Y \in S$ 
  unfolding eventually_finite_subsets_at_top by metis
define  $n$  where  $n \equiv \text{Max } X + 1$ 
show ?thesis
proof (subst eventually_sequentially, standard, safe)
  fix  $m$  assume  $Hm: n \leq m$ 
  moreover have  $x \in X \implies x < n$  for  $x$ 
    unfolding  $n\_def$  using Max_ge [OF hX(1), of x] by auto
  ultimately show  $\text{sum } f (\{..<m\} \cap A) \in S$ 
    using  $hX(2)$  by (intro hY, auto) (metis order.strict_trans2)
  qed
qed
thus ?thesis unfolding has_subsum_def sums_def tendsto_def
  by (simp add: sum.inter_restrict [symmetric])
qed

```

```

unbundle no_pnt_notation
end
theory Perron_Formula
imports
  PNT_Remainder_Library
  PNT_Subsummable
begin
unbundle pnt_notation

```

5 Perron's formula

This version of Perron's theorem is referenced to: *Perron's Formula and the Prime Number Theorem for Automorphic L-Functions*, Jianya Liu, Y. Ye

A contour integral estimation lemma that will be used both in proof of Perron's formula and the prime number theorem.

lemma *perron_aux_3'*:

```

fixes  $f :: \text{complex} \Rightarrow \text{complex}$  and  $a b B T :: \text{real}$ 
assumes  $Ha: 0 < a$  and  $Hb: 0 < b$  and  $hT: 0 < T$ 
  and  $Hf: \bigwedge t. t \in \{-T..T\} \implies \|f (\text{Complex } b t)\| \leq B$ 
  and  $Hf': (\lambda s. f s * a \text{ powr } s / s) \text{ contour\_integrable\_on } (\text{linepath } (\text{Complex } b (-T)) (\text{Complex } b T))$ 
shows  $\|1 / (2 * \text{pi} * i) * \text{contour\_integral } (\text{linepath } (\text{Complex } b (-T)) (\text{Complex } b T)) (\lambda s. f s * a \text{ powr } s / s)\|$ 
   $\leq B * a \text{ powr } b * \ln (1 + T / b)$ 

```

proof –

```

define path where  $\text{path} \equiv \text{linepath } (\text{Complex } b (-T)) (\text{Complex } b T)$ 
define  $t'$  where  $t' t \equiv \text{Complex } (\text{Re } (\text{Complex } b (-T))) t$  for  $t$ 
define  $g$  where  $g t \equiv f (\text{Complex } b t) * a \text{ powr } (\text{Complex } b t) / \text{Complex } b t * i$  for  $t$ 
have  $\|f (\text{Complex } b 0)\| \leq B$  using  $hT$  by (auto intro: Hf [of 0])
hence  $hB: 0 \leq B$  using  $hT$  by (smt (verit) norm_ge_zero)
have  $((\lambda t. f (t' t) * a \text{ powr } (t' t) / (t' t) * i)$ 
   $\text{has\_integral } \text{contour\_integral } \text{path } (\lambda s. f s * a \text{ powr } s / s)) \{ \text{Im } (\text{Complex } b (-T)) .. \text{Im } (\text{Complex } b T) \}$ 
  unfolding  $t'\_def$  using  $hT$ 
  by (intro integral_linepath_same_Re, unfold path_def)

```

```

(auto intro: has_contour_integral_integral Hf')
hence h_int: (g has_integral contour_integral path (λs. f s * a powr s / s)) {-T..T}
  unfolding g_def t'_def by auto
hence int: g integrable_on {-T..T} by (rule has_integral_integrable)
have contour_integral path (λs. f s * a powr s / s) = integral {-T..T} g
  using h_int by (rule integral_unique [symmetric])
also have ||...|| ≤ integral {-T..T} (λt. 2 * B * a powr b / (b + |t|))
proof (rule integral_norm_bound_integral, goal_cases)
  case 1 from int show ?case .
  case 2 show ?case
    by (intro integrable_continuous_interval continuous_intros)
      (use Hb in auto)
next
fix t assume *: t ∈ {-T..T}
have (b + |t|)2 - 4 * (b2 + t2) = - 3 * (b - |t|)2 + - 4 * b * |t|
  by (simp add: field_simps power2_eq_square)
also have ... ≤ 0 using Hb by (intro add_nonpos_nonpos) auto
finally have (b + |t|)2 - 4 * (b2 + t2) ≤ 0 .
hence b + |t| ≤ 2 * ||Complex b t||
  unfolding cmod_def by (auto intro: power2_le_imp_le)
hence a powr b / ||Complex b t|| ≤ a powr b / ((b + |t|) / 2)
  using Hb by (intro divide_left_mono) (auto intro!: mult_pos_pos)
hence a powr b / ||Complex b t|| * ||f (Complex b t)|| ≤ a powr b / ((b + |t|) / 2) * B
  by (insert Hf [OF *], rule mult_mono) (use Hb in auto)
thus ||g t|| ≤ 2 * B * a powr b / (b + |t|)
  unfolding g_def
  by (auto simp add: norm_mult norm_divide)
    (subst norm_powr_real_powr, insert Ha, auto simp add: mult_ac)
qed
also have ... = 2 * B * a powr b * integral {-T..T} (λt. 1 / (b + |t|))
  by (subst divide_inverse, subst integral_mult_right) (simp add: inverse_eq_divide)
also have ... = 4 * B * a powr b * integral {0..T} (λt. 1 / (b + |t|))
proof -
  let ?f = λt. 1 / (b + |t|)
  have integral {-T..0} ?f + integral {0..T} ?f = integral {-T..T} ?f
    by (intro Henstock_Kurzweil_Integration.integral_combine
      integrable_continuous_interval continuous_intros)
      (use Hb hT in auto)
  moreover have integral {-T..-0} (λt. ?f (-t)) = integral {0..T} ?f
    by (rule Henstock_Kurzweil_Integration.integral_reflect_real)
  hence integral {-T..0} ?f = integral {0..T} ?f by auto
  ultimately show ?thesis by auto
qed
also have ... = 4 * B * a powr b * ln (1 + T / b)
proof -
  have ((λt. 1 / (b + |t|)) has_integral (ln (b + T) - ln (b + 0))) {0..T}
  proof (rule fundamental_theorem_of_calculus, goal_cases)
    case 1 show ?case using hT by auto
  next
  fix x assume *: x ∈ {0..T}
  have ((λx. ln (b + x)) has_real_derivative 1 / (b + x) * (0 + 1)) (at x within {0..T})
    by (intro derivative_intros) (use Hb * in auto)
  thus ((λx. ln (b + x)) has_vector_derivative 1 / (b + |x|)) (at x within {0..T})
    using * by (subst has_real_derivative_iff_has_vector_derivative [symmetric]) auto
qed

```

moreover have $\ln (b + T) - \ln (b + 0) = \ln (1 + T / b)$
using Hb hT **by** (*subst ln_div [symmetric]*) (*auto simp add: field_simps*)
ultimately show *?thesis* **by** *auto*
qed
finally have $\|1 / (2 * \pi * i) * \text{contour_integral path } (\lambda s. f s * a \text{ powr } s / s)\|$
 $\leq 1 / (2 * \pi) * 4 * B * a \text{ powr } b * \ln (1 + T / b)$
by (*simp add: norm_divide norm_mult field_simps*)
also have $\dots \leq 1 * B * a \text{ powr } b * \ln (1 + T / b)$
proof –
have $1 / (2 * \pi) * 4 \leq 1$ **using** *pi_gt3* **by** *auto*
thus *?thesis* **by** (*intro mult_right_mono*) (*use hT Hb hB in auto*)
qed
finally show *?thesis* **unfolding** *path_def* **by** *auto*
qed

locale *perron_locale* =
fixes $b B H T x :: \text{real}$ **and** $f :: \text{complex fds}$
assumes $Hb: 0 < b$ **and** $hT: b \leq T$
and $Hb': \text{abs_conv_abscissa } f < b$
and $hH: 2 \leq H$ **and** $hH': b + 1 \leq H$ **and** $Hx: 0 < x$
and $hB: (\sum 'n \geq 1. \|f_{ds_nth } f n\| / n \text{ nat_powr } b) \leq B$
begin
definition r **where** $r a \equiv$
if $a \neq 1$ *then* $\min (1 / (2 * T * |\ln a|)) (2 + \ln (T / b))$
else $(2 + \ln (T / b))$
definition *path* **where** $\text{path} \equiv \text{linepath } (\text{Complex } b (-T)) (\text{Complex } b T)$
definition *img_path* **where** $\text{img_path} \equiv \text{path_image path}$
definition σ_a **where** $\sigma_a \equiv \text{abs_conv_abscissa } f$
definition *region* **where** $\text{region} = \{n :: \text{nat}. x - x / H \leq n \wedge n \leq x + x / H\}$
definition F **where** $F (a :: \text{real}) \equiv$
 $1 / (2 * \pi * i) * \text{contour_integral path } (\lambda s. a \text{ powr } s / s) - (\text{if } 1 \leq a \text{ then } 1 \text{ else } 0)$
definition F' **where** $F' (n :: \text{nat}) \equiv F (x / n)$

lemma $hT': 0 < T$ **using** Hb hT **by** *auto*
lemma *cond*: $0 < b$ $b \leq T$ $0 < T$ **using** Hb hT hT' **by** *auto*

lemma *perron_integrable*:
assumes $(0 :: \text{real}) < a$
shows $(\lambda s. a \text{ powr } s / s)$ *contour_integrable_on* (*linepath* (*Complex* b $(-T)$)) (*Complex* b T)
using *cond* *assms*
by (*intro contour_integrable_continuous_linepath continuous_intros*)
(auto simp add: closed_segment_def legacy_Complex_simps field_simps)

lemma *perron_aux_1'*:
fixes $U :: \text{real}$
assumes $hU: 0 < U$ **and** $Ha: 1 < a$
shows $\|F a\| \leq 1 / \pi * a \text{ powr } b / (T * |\ln a|) + a \text{ powr } -U * T / (\pi * U)$
proof –
define f **where** $f \equiv \lambda s :: \text{complex}. a \text{ powr } s / s$
note $\text{assms}' = \text{cond } \text{assms}$ *this*
define P_1 **where** $P_1 \equiv \text{linepath } (\text{Complex } (-U) (-T)) (\text{Complex } b (-T))$
define P_2 **where** $P_2 \equiv \text{linepath } (\text{Complex } b (-T)) (\text{Complex } b T)$
define P_3 **where** $P_3 \equiv \text{linepath } (\text{Complex } b T) (\text{Complex } (-U) T)$
define P_4 **where** $P_4 \equiv \text{linepath } (\text{Complex } (-U) T) (\text{Complex } (-U) (-T))$
define P **where** $P \equiv P_1 +++ P_2 +++ P_3 +++ P_4$

define $I_1 I_2 I_3 I_4$ **where**
 $I_1 \equiv \text{contour_integral } P_1 f$ **and** $I_2 \equiv \text{contour_integral } P_2 f$ **and**
 $I_3 \equiv \text{contour_integral } P_3 f$ **and** $I_4 \equiv \text{contour_integral } P_4 f$
define rpath **where** $\text{rpath} \equiv \text{rectpath } (\text{Complex } (-U) (-T)) (\text{Complex } b T)$
note $P_defs = P_def P_1_def P_2_def P_3_def P_4_def$
note $I_defs = I_1_def I_2_def I_3_def I_4_def$
have $1: \bigwedge A B x. A \subseteq B \implies x \notin A \implies A \subseteq B - \{x\}$ **by** *auto*
have $\text{path_image } (\text{rectpath } (\text{Complex } (-U) (-T)) (\text{Complex } b T))$
 $\subseteq \text{cbox } (\text{Complex } (-U) (-T)) (\text{Complex } b T) - \{0\}$
using *assms'*
by (*intro 1 path_image_rectpath_subset_cbox*)
(*auto simp add: path_image_rectpath*)
moreover have $0 \in \text{box } (\text{Complex } (-U) (-T)) (\text{Complex } b T)$
using *assms'* **by** (*simp add: mem_box Basis_complex_def*)
ultimately have
(*($\lambda s. a \text{ powr } s / (s - 0)$) has_contour_integral*
 $2 * \text{pi} * i * \text{winding_number } \text{rpath } 0 * a \text{ powr } (0 :: \text{complex})) \text{rpath}$
 $\text{winding_number } \text{rpath } 0 = 1$
unfolding rpath_def
by (*intro Cauchy_integral_formula_convex_simple*
[**where** $S = \text{cbox } (\text{Complex } (-U) (-T)) (\text{Complex } b T)$])
(*auto intro!: assms' holomorphic_on_powr_right winding_number_rectpath*
simp add: mem_box Basis_complex_def)
hence (*f has_contour_integral* $2 * \text{pi} * i$) rpath **unfolding** f_def **using** H_a **by** *auto*
hence $2: (f \text{ has_contour_integral } 2 * \text{pi} * i) P$
unfolding $\text{rpath_def } P_defs \text{rectpath_def } \text{Let_def}$ **by** *simp*
hence $f \text{ contour_integrable_on } P$ **by** (*intro has_contour_integral_integrable*) (*use 2 in auto*)
hence $3: f \text{ contour_integrable_on } P_1 f \text{ contour_integrable_on } P_2$
 $f \text{ contour_integrable_on } P_3 f \text{ contour_integrable_on } P_4$ **unfolding** P_defs **by** *auto*
from 2 have $I_1 + I_2 + I_3 + I_4 = 2 * \text{pi} * i$ **unfolding** $P_defs I_defs$ **by** (*rule has_chain_integral_chain_integral*)
hence $I_2 - 2 * \text{pi} * i = -(I_1 + I_3 + I_4)$ **by** (*simp add: field_simps*)
hence $\|I_2 - 2 * \text{pi} * i\| = \|(I_1 + I_3 + I_4)\|$ **by** *auto*
also have $\dots = \|I_1 + I_3 + I_4\|$ **by** (*rule norm_minus_cancel*)
also have $\dots \leq \|I_1 + I_3\| + \|I_4\|$ **by** (*rule norm_triangle_ineq*)
also have $\dots \leq \|I_1\| + \|I_3\| + \|I_4\|$ **using** *norm_triangle_ineq* **by** *auto*
finally have $*$: $\|I_2 - 2 * \text{pi} * i\| \leq \|I_1\| + \|I_3\| + \|I_4\|$.
have $I_2_val: \|I_2 / (2 * \text{pi} * i) - 1\| \leq 1 / (2 * \text{pi}) * (\|I_1\| + \|I_3\| + \|I_4\|)$
proof -
have $I_2 - 2 * \text{pi} * i = (I_2 / (2 * \text{pi} * i) - 1) * (2 * \text{pi} * i)$ **by** (*auto simp add: field_simps*)
hence $\|I_2 - 2 * \text{pi} * i\| = \|(I_2 / (2 * \text{pi} * i) - 1) * (2 * \text{pi} * i)\|$ **by** *auto*
also have $\dots = \|I_2 / (2 * \text{pi} * i) - 1\| * (2 * \text{pi})$ **by** (*auto simp add: norm_mult*)
finally have $\|I_2 / (2 * \text{pi} * i) - 1\| = 1 / (2 * \text{pi}) * \|I_2 - 2 * \text{pi} * i\|$ **by** *auto*
also have $\dots \leq 1 / (2 * \text{pi}) * (\|I_1\| + \|I_3\| + \|I_4\|)$
using $*$ **by** (*subst mult_le_cancel_left_pos*) *auto*
finally show *?thesis* .
qed
define Q **where** $Q t \equiv \text{linepath } (\text{Complex } (-U) t) (\text{Complex } b t)$ **for** t
define g **where** $g t \equiv \text{contour_integral } (Q t) f$ **for** t
have $Q_1: (f \text{ has_contour_integral } I_1) (Q (-T))$
using $3(1)$ **unfolding** $P_1_def I_1_def Q_def$
by (*rule has_contour_integral_integral*)
have $Q_2: (f \text{ has_contour_integral } -I_3) (Q T)$
using $3(3)$ **unfolding** $P_3_def I_3_def Q_def$
by (*subst contour_integral_reversepath [symmetric]*,
auto intro!: has_contour_integral_integral)

```

    (subst contour_integrable_reversepath_eq [symmetric], auto)
have subst_I1_I3: I1 = g (- T) I3 = - g T
  using Q_1 Q_2 unfolding g_def by (auto simp add: contour_integral_unique)
have g_bound: ||g t|| ≤ a powr b / (T * |ln a|)
  when Ht: |t| = T for t
proof -
  have (f has_contour_integral g t) (Q t) proof -
    consider t = T | t = -T using Ht by fastforce
    hence f_contour_integrable_on Q t using Q_1 Q_2 by (metis has_contour_integral_integrable)
    thus ?thesis unfolding g_def by (rule has_contour_integral_integral)
  qed
  hence ((λx. a powr (x + Im (Complex (-U) t) * i) / (x + Im (Complex (-U) t) * i)) has_integral (g
t))
    {Re (Complex (-U) t) .. Re (Complex b t)}
  unfolding Q_def f_def
  by (subst has_contour_integral_linepath_same_Im_iff [symmetric])
    (use hU Hb in auto)
  hence *: ((λx. a powr (x + t * i) / (x + t * i)) has_integral g t) {-U..b} by auto
  hence ||g t|| = ||integral {-U..b} (λx. a powr (x + t * i) / (x + t * i))|| by (auto simp add: inte-
gral_unique)
  also have ... ≤ integral {-U..b} (λx. a powr x / T)
  proof (rule integral_norm_bound_integral)
    show (λx. a powr (x + t * i) / (x + t * i)) integrable_on {-U..b} using * by auto
    have (λx. a powr x / (of_real T)) integrable_on {-U..b}
      by (intro iffD2 [OF integrable_on_cdivide_iff] powr_integrable) (use hU Ha Hb hT' in auto)
    thus (λx. a powr x / T) integrable_on {-U..b} by auto
  next
    fix x assume x ∈ {-U..b}
    have ||a powr (x + t * i)|| = Re a powr Re (x + t * i) by (rule norm_powr_real_powr) (use Ha in
auto)
    also have ... = a powr x by auto
    finally have *: ||a powr (x + t * i)|| = a powr x .
    have T = |Im (x + t * i)| using Ht by auto
    also have ... ≤ ||x + t * i|| by (rule abs_Im_le_cmod)
    finally have T ≤ ||x + t * i|| .
    with * show ||a powr (x + t * i) / (x + t * i)|| ≤ a powr x / T
      by (subst norm_divide) (rule frac_le, use assms' in auto)
  qed
  also have ... = integral {-U..b} (λx. a powr x) / T by auto
  also have ... ≤ a powr b / (T * |ln a|)
  proof -
    have integral {-U..b} (λx. a powr x) ≤ a powr b / |ln a|
      by (rule powr_integral_bound_gt_1) (use hU Ha Hb in auto)
    thus ?thesis using hT' by (auto simp add: field_simps)
  qed
  finally show ?thesis .
qed
have ||I4|| ≤ a powr -U / U * ||Complex (- U) (- T) - Complex (- U) T||
proof -
  have f_contour_integrable_on P4 by (rule 3)
  moreover have 0 ≤ a powr -U / U using hU by auto
  moreover have ||f z|| ≤ a powr -U / U
    when *: z ∈ closed_segment (Complex (- U) T) (Complex (- U) (- T)) for z
  proof -
    from * have Re_z: Re z = - U

```

unfolding *closed_segment_def*
by (*auto simp add: legacy_Complex_simps field_simps*)
hence $U = |\operatorname{Re} z|$ **using** *hU* **by** *auto*
also have $\dots \leq \|z\|$ **by** (*rule abs_Re_le_cmod*)
finally have *zmod*: $U \leq \|z\|$.
have $\|f z\| = \|a \operatorname{powr} z\| / \|z\|$ **unfolding** *f_def* **by** (*rule norm_divide*)
also have $\dots \leq a \operatorname{powr} - U / U$
by (*subst norm_powr_real_powr, use Ha in auto*)
(*rule frac_le, use hU Re_z zmod in auto*)
finally show *?thesis* .
qed
ultimately show *?thesis* **unfolding** *I4_def P4_def* **by** (*rule contour_integral_bound_linepath*)
qed
also have $\dots = a \operatorname{powr} -U / U * (2 * T)$
proof –
have $\sqrt{(2 * T)^2} = |2 * T|$ **by** (*rule real_sqrt_abs*)
thus *?thesis* **using** *hT'* **by** (*auto simp add: field_simps legacy_Complex_simps*)
qed
finally have *I4_bound*: $\|I_4\| \leq a \operatorname{powr} -U / U * (2 * T)$.
have $\|I_2 / (2 * \pi * i) - 1\| \leq 1 / (2 * \pi) * (\|g(-T)\| + \|-g T\| + \|I_4\|)$
using *I2_val subst_I1_I3* **by** *auto*
also have $\dots \leq 1 / (2 * \pi) * (2 * a \operatorname{powr} b / (T * |\ln a|) + a \operatorname{powr} -U / U * (2 * T))$
proof –
have $\|g T\| \leq a \operatorname{powr} b / (T * |\ln a|)$
 $\|g(-T)\| \leq a \operatorname{powr} b / (T * |\ln a|)$
using *hT'* **by** (*auto intro: g_bound*)
hence $\|g(-T)\| + \|-g T\| + \|I_4\| \leq 2 * a \operatorname{powr} b / (T * |\ln a|) + a \operatorname{powr} -U / U * (2 * T)$
using *I4_bound* **by** *auto*
thus *?thesis* **by** (*auto simp add: field_simps*)
qed
also have $\dots = 1 / \pi * a \operatorname{powr} b / (T * |\ln a|) + a \operatorname{powr} -U * T / (\pi * U)$
using *hT'* **by** (*auto simp add: field_simps*)
finally show *?thesis*
using *Ha* **unfolding** *I2_def P2_def f_def F_def path_def* **by** *auto*
qed

lemma *perron_aux_1*:

assumes *Ha*: $1 < a$
shows $\|F a\| \leq 1 / \pi * a \operatorname{powr} b / (T * |\ln a|)$ (**is** $_ \leq ?x$)

proof –

let $?y = \lambda U :: \operatorname{real}. a \operatorname{powr} -U * T / (\pi * U)$
have $((\lambda U :: \operatorname{real}. ?x) \longrightarrow ?x)$ **at_top** **by** *auto*
moreover have $((\lambda U. ?y U) \longrightarrow 0)$ **at_top** **using** *Ha* **by** *real_asymp*
ultimately have $((\lambda U. ?x + ?y U) \longrightarrow ?x + 0)$ **at_top** **by** (*rule tendsto_add*)
hence $((\lambda U. ?x + ?y U) \longrightarrow ?x)$ **at_top** **by** *auto*
moreover have $\|F a\| \leq ?x + ?y U$ **when** *hU*: $0 < U$ **for** *U*
by (*subst perron_aux_1' [OF hU Ha], standard*)
hence $\forall_F U$ **in** *at_top*. $\|F a\| \leq ?x + ?y U$
by (*rule eventually_at_top_linorderI'*)
ultimately show *?thesis*
by (*intro tendsto_lowerbound*) *auto*

qed

lemma *perron_aux_2'*:

fixes *U* **::** *real*

assumes $hU: 0 < U b < U$ **and** $Ha: 0 < a \wedge a < 1$
shows $\|F a\| \leq 1 / \pi i * a \text{ powr } b / (T * |\ln a|) + a \text{ powr } U * T / (\pi i * U)$
proof –
define f **where** $f \equiv \lambda s :: \text{complex. } a \text{ powr } s / s$
note $\text{assms}' = \text{cond assms } hU$
define P_1 **where** $P_1 \equiv \text{linepath } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ (-T))$
define P_2 **where** $P_2 \equiv \text{linepath } (\text{Complex } U \ (-T)) \ (\text{Complex } U \ T)$
define P_3 **where** $P_3 \equiv \text{linepath } (\text{Complex } U \ T) \ (\text{Complex } b \ T)$
define P_4 **where** $P_4 \equiv \text{linepath } (\text{Complex } b \ T) \ (\text{Complex } b \ (-T))$
define P **where** $P \equiv P_1 +++ P_2 +++ P_3 +++ P_4$
define $I_1 \ I_2 \ I_3 \ I_4$ **where**
 $I_1 \equiv \text{contour_integral } P_1 \ f$ **and** $I_2 \equiv \text{contour_integral } P_2 \ f$ **and**
 $I_3 \equiv \text{contour_integral } P_3 \ f$ **and** $I_4 \equiv \text{contour_integral } P_4 \ f$
define $rpath$ **where** $rpath \equiv \text{rectpath } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ T)$
note $P_defs = P_def \ P_1_def \ P_2_def \ P_3_def \ P_4_def$
note $I_defs = I_1_def \ I_2_def \ I_3_def \ I_4_def$
have $\text{path_image } (\text{rectpath } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ T)) \subseteq \text{cbox } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ T)$
by $(\text{intro path_image_rectpath_subset_cbox})$ $(\text{use assms}' \ \text{in } \text{auto})$
moreover **have** $0 \notin \text{cbox } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ T)$
using Hb **unfolding** cbox_def **by** $(\text{auto simp add: Basis_complex_def})$
ultimately **have** $(\lambda s. a \text{ powr } s / (s - 0)) \ \text{has_contour_integral } 0$ $rpath$
unfolding $rpath_def$
by $(\text{intro Cauchy_theorem_convex_simple}$
 $[\text{where } S = \text{cbox } (\text{Complex } b \ (-T)) \ (\text{Complex } U \ T)])$
 $(\text{auto intro!: holomorphic_on_powr_right holomorphic_on_divide})$
hence $(f \ \text{has_contour_integral } 0) \ rpath$ **unfolding** f_def **using** Ha **by** auto
hence $1: (f \ \text{has_contour_integral } 0) \ P$ **unfolding** $rpath_def \ P_defs \ \text{rectpath_def}$ Let_def **by** simp
hence $f \ \text{contour_integrable_on } P$ **by** $(\text{intro has_contour_integral_integrable})$ $(\text{use } 1 \ \text{in } \text{auto})$
hence $2: f \ \text{contour_integrable_on } P_1 \ f \ \text{contour_integrable_on } P_2$
 $f \ \text{contour_integrable_on } P_3 \ f \ \text{contour_integrable_on } P_4$ **unfolding** P_defs **by** auto
from 1 **have** $I_1 + I_2 + I_3 + I_4 = 0$ **unfolding** $P_defs \ I_defs$ **by** $(\text{rule has_chain_integral_chain_integral4})$
hence $I_4 = - (I_1 + I_2 + I_3)$ **by** $(\text{metis neg_eq_iff_add_eq_0})$
hence $\|I_4\| = \| - (I_1 + I_2 + I_3) \|$ **by** auto
also **have** $\dots = \|I_1 + I_2 + I_3\|$ **by** $(\text{rule norm_minus_cancel})$
also **have** $\dots \leq \|I_1 + I_2\| + \|I_3\|$ **by** $(\text{rule norm_triangle_ineq})$
also **have** $\dots \leq \|I_1\| + \|I_2\| + \|I_3\|$ **using** $\text{norm_triangle_ineq}$ **by** auto
finally **have** $\|I_4\| \leq \|I_1\| + \|I_2\| + \|I_3\|$.
hence $I_4_val: \|I_4 / (2 * \pi i * i)\| \leq 1 / (2 * \pi i) * (\|I_1\| + \|I_2\| + \|I_3\|)$
by $(\text{auto simp add: norm_divide norm_mult_field_simps})$
define Q **where** $Q \ t \equiv \text{linepath } (\text{Complex } b \ t) \ (\text{Complex } U \ t)$ **for** t
define g **where** $g \ t \equiv \text{contour_integral } (Q \ t) \ f$ **for** t
have $Q_1: (f \ \text{has_contour_integral } I_1) \ (Q \ (-T))$
using $2(1)$ **unfolding** $P_1_def \ I_1_def \ Q_def$
by $(\text{rule has_contour_integral_integral})$
have $Q_2: (f \ \text{has_contour_integral } -I_3) \ (Q \ T)$
using $2(3)$ **unfolding** $P_3_def \ I_3_def \ Q_def$
by $(\text{subst contour_integral_reversepath [symmetric],}$
 $\text{auto intro!: has_contour_integral_integral})$
 $(\text{subst contour_integrable_reversepath_eq [symmetric], auto})$
have $\text{subst } I_1 \ I_3: I_1 = g \ (-T) \ I_3 = - g \ T$
using $Q_1 \ Q_2$ **unfolding** g_def **by** $(\text{auto simp add: contour_integral_unique})$
have $g_bound: \|g \ t\| \leq a \text{ powr } b / (T * |\ln a|)$
when $Ht: |t| = T$ **for** t
proof –

have (f has_contour_integral g t) (Q t) **proof** –
consider $t = T \mid t = -T$ **using** Ht **by** *fastforce*
hence f contour_integrable_on Q t **using** Q_1 Q_2 **by** (*metis* has_contour_integral_integrable)
thus ?thesis **unfolding** g_def **by** (*rule* has_contour_integral_integral)
qed
hence ($(\lambda x. a$ powr $(x + \text{Im} (\text{Complex } b$ $t) * i) / (x + \text{Im} (\text{Complex } b$ $t) * i))$ has_integral $(g$ t)
 $\{ \text{Re} (\text{Complex } b$ $t) .. \text{Re} (\text{Complex } U$ $t) \}$
unfolding Q_def f_def
by (*subst* has_contour_integral_linepath_same_Im_iff [*symmetric*])
(use *assms'* **in** *auto*)
hence *: ($(\lambda x. a$ powr $(x + t * i) / (x + t * i))$ has_integral g t) $\{b..U\}$ **by** *auto*
hence $\|g$ $t\| = \|integral \{b..U\} (\lambda x. a$ powr $(x + t * i) / (x + t * i))\|$ **by** (*auto* *simp* *add*: *integral_unique*)
also have $\dots \leq integral \{b..U\} (\lambda x. a$ powr $x / T)$
proof (*rule* *integral_norm_bound_integral*)
show $(\lambda x. a$ powr $(x + t * i) / (x + t * i))$ integrable_on $\{b..U\}$ **using** * **by** *auto*
have $(\lambda x. a$ powr $x / (\text{of_real } T))$ integrable_on $\{b..U\}$
by (*intro* *iffD2* [*OF* *integrable_on_cdivide_iff*] *powr_integrable*) (*use* *assms'* **in** *auto*)
thus $(\lambda x. a$ powr $x / T)$ integrable_on $\{b..U\}$ **by** *auto*
next
fix x **assume** $x \in \{b..U\}$
have $\|a$ powr $(x + t * i)\| = \text{Re } a$ powr $\text{Re} (x + t * i)$ **by** (*rule* *norm_powr_real_powr*) (*use* Ha **in** *auto*)
also have $\dots = a$ powr x **by** *auto*
finally have 1: $\|a$ powr $(x + t * i)\| = a$ powr x .
have $T = |\text{Im} (x + t * i)|$ **using** Ht **by** *auto*
also have $\dots \leq \|x + t * i\|$ **by** (*rule* *abs_Im_le_cmod*)
finally have 2: $T \leq \|x + t * i\|$.
from 1 2 **show** $\|a$ powr $(x + t * i) / (x + t * i)\| \leq a$ powr x / T
by (*subst* *norm_divide*) (*rule* *frac_le*, *use* hT' **in** *auto*)
qed
also have $\dots = integral \{b..U\} (\lambda x. a$ powr $x) / T$ **by** *auto*
also have $\dots \leq a$ powr $b / (T * |\ln a|)$
proof –
have $integral \{b..U\} (\lambda x. a$ powr $x) \leq a$ powr $b / |\ln a|$
by (*rule* *powr_integral_bound_lt_1*) (*use* *assms'* **in** *auto*)
thus ?thesis **using** hT' **by** (*auto* *simp* *add*: *field_simps*)
qed
finally show ?thesis .
qed
have $\|I_2\| \leq a$ powr $U / U * \|Complex$ U $T - Complex$ U $(- T)\|$
proof –
have f contour_integrable_on P_2 **by** (*rule* 2)
moreover have $0 \leq a$ powr U / U **using** hU **by** *auto*
moreover have $\|f$ $z\| \leq a$ powr U / U
when *: $z \in \text{closed_segment} (\text{Complex } U$ $(- T)) (\text{Complex } U$ $T)$ **for** z
proof –
from * **have** $\text{Re } z$: $\text{Re } z = U$
unfolding *closed_segment_def*
by (*auto* *simp* *add*: *legacy_Complex_simps* *field_simps*)
hence $U = |\text{Re } z|$ **using** hU **by** *auto*
also have $\dots \leq \|z\|$ **by** (*rule* *abs_Re_le_cmod*)
finally have $zmod$: $U \leq \|z\|$.
have $\|f$ $z\| = \|a$ powr $z\| / \|z\|$ **unfolding** f_def **by** (*rule* *norm_divide*)
also have $\dots \leq a$ powr U / U
by (*subst* *norm_powr_real_powr*, *use* Ha **in** *auto*)

(rule frac_le, use hU Re_z zmod in auto)
 finally show ?thesis .
 qed
 ultimately show ?thesis unfolding I2_def P2_def by (rule contour_integral_bound_linepath)
 qed
 also have ... ≤ a powr U / U * (2 * T)
 proof -
 have sqrt ((2 * T)²) = |2 * T| by (rule real_sqrt_abs)
 thus ?thesis using hT' by (simp add: field_simps legacy_Complex_simps)
 qed
 finally have I2_bound: ||I2|| ≤ a powr U / U * (2 * T) .
 have ||I4 / (2 * pi * i)|| ≤ 1 / (2*pi) * (||g (- T)|| + ||I2|| + ||- g T||)
 using I4_val subst_I1_I3 by auto
 also have ... ≤ 1 / (2*pi) * (2 * a powr b / (T * |ln a|) + a powr U / U * (2*T))
 proof -
 have ||g T|| ≤ a powr b / (T * |ln a|)
 ||g (- T)|| ≤ a powr b / (T * |ln a|)
 using hT' by (auto intro: g_bound)
 hence ||g (- T)|| + ||- g T|| + ||I2|| ≤ 2 * a powr b / (T * |ln a|) + a powr U / U * (2*T)
 using I2_bound by auto
 thus ?thesis by (auto simp add: field_simps)
 qed
 also have ... = 1 / pi * a powr b / (T * |ln a|) + a powr U * T / (pi * U)
 using hT' by (auto simp add: field_simps)
 finally have ||1 / (2 * pi * i) * contour_integral (reversepath P4) f||
 ≤ 1 / pi * a powr b / (T * |ln a|) + a powr U * T / (pi * U)
 unfolding I4_def P4_def by (subst contour_integral_reversepath) auto
 thus ?thesis using Ha unfolding I4_def P4_def f_def F_def path_def by auto
 qed

lemma perron_aux_2:

assumes Ha: 0 < a ∧ a < 1
 shows ||F a|| ≤ 1 / pi * a powr b / (T * |ln a|) (is _ ≤ ?x)
 proof -
 let ?y = λU :: real. a powr U * T / (pi * U)
 have ((λU :: real. ?x) → ?x) at_top by auto
 moreover have ((λU. ?y U) → 0) at_top using Ha by real_asymp
 ultimately have ((λU. ?x + ?y U) → ?x + 0) at_top by (rule tendsto_add)
 hence ((λU. ?x + ?y U) → ?x) at_top by auto
 moreover have ||F a|| ≤ ?x + ?y U when hU: 0 < U b < U for U
 by (subst perron_aux_2' [OF hU Ha], standard)
 hence ∀_F U in at_top. ||F a|| ≤ ?x + ?y U
 by (rule eventually_at_top_linorderI') (use Hb in auto)
 ultimately show ?thesis
 by (intro tendsto_lowerbound) auto
 qed

lemma perron_aux_3:

assumes Ha: 0 < a
 shows ||1 / (2 * pi * i) * contour_integral path (λs. a powr s / s)|| ≤ a powr b * ln (1 + T / b)
 proof -
 have ||1 / (2 * pi * i) * contour_integral (linepath (Complex b (-T)) (Complex b T)) (λs. 1 * a powr s / s)||
 ≤ 1 * a powr b * ln (1 + T / b)
 by (rule perron_aux_3') (auto intro: Ha cond_perron_integrable)

thus *?thesis* unfolding *path_def* by *auto*
qed

lemma *perron_aux'*:

assumes *Ha*: $0 < a$

shows $\|F a\| \leq a \text{ powr } b * r a$

proof –

note *assms'* = *assms cond*

define *P* where $P \equiv 1 / (2 * \pi * i) * \text{contour_integral path } (\lambda s. a \text{ powr } s / s)$

have *lm_1*: $1 + \ln (1 + T / b) \leq 2 + \ln (T / b)$

proof –

have $1 \leq T / b$ using *hT Hb* by *auto*

hence $1 + T / b \leq 2 * (T / b)$ by *auto*

hence $\ln (1 + T / b) \leq \ln 2 + \ln (T / b)$ by (*subst ln_mult [symmetric]*) *auto*

thus *?thesis* using *ln_2_less_1* by *auto*

qed

have *: $\|F a\| \leq a \text{ powr } b * (2 + \ln (T / b))$

proof (*cases* $1 \leq a$)

assume *Ha'*: $1 \leq a$

have $\|P - 1\| \leq \|P\| + 1$ by (*simp add: norm_triangle_le_diff*)

also have $\dots \leq a \text{ powr } b * \ln (1 + T / b) + 1$

proof –

have $\|P\| \leq a \text{ powr } b * \ln (1 + T / b)$

unfolding *P_def* by (*intro perron_aux_3 assms'*)

thus *?thesis* by *auto*

qed

also have $\dots \leq a \text{ powr } b * (2 + \ln (T / b))$

proof –

have $1 = a \text{ powr } 0$ using *Ha'* by *auto*

also have $a \text{ powr } 0 \leq a \text{ powr } b$ using *Ha' Hb* by (*intro powr_mono*) *auto*

finally have $a \text{ powr } b * \ln (1 + T / b) + 1 \leq a \text{ powr } b * (1 + \ln (1 + T / b))$

by (*auto simp add: algebra_simps*)

also have $\dots \leq a \text{ powr } b * (2 + \ln (T / b))$ using *Ha' lm_1* by *auto*

finally show *?thesis* .

qed

finally show *?thesis* using *Ha'* unfolding *F_def P_def* by *auto*

next

assume *Ha'*: $\neg 1 \leq a$

hence $\|P\| \leq a \text{ powr } b * \ln (1 + T / b)$

unfolding *P_def* by (*intro perron_aux_3 assms'*)

also have $\dots \leq a \text{ powr } b * (2 + \ln (T / b))$

by (*rule mult_left_mono*) (*use lm_1 in auto*)

finally show *?thesis* using *Ha'* unfolding *F_def P_def* by *auto*

qed

consider $0 < a \wedge a \neq 1 \mid a = 1$ using *Ha* by *linarith*

thus *?thesis* proof *cases*

define *c* where $c = 1 / 2 * a \text{ powr } b / (T * |\ln a|)$

assume *Ha'*: $0 < a \wedge a \neq 1$

hence $(0 < a \wedge a < 1) \vee a > 1$ by *auto*

hence $\|F a\| \leq 1 / \pi * a \text{ powr } b / (T * |\ln a|)$

using *perron_aux_1 perron_aux_2* by *auto*

also have $\dots \leq c$ unfolding *c_def*

using *Ha' hT' pi_gt3* by (*auto simp add: field_simps*)

finally have $\|F a\| \leq c$.

hence $\|F a\| \leq \min c (a \text{ powr } b * (2 + \ln (T / b)))$ using * by *auto*

also have ... = a powr b * r a
 unfolding r_def c_def using Ha' by auto (subst min_mult_distrib_left, auto)
 finally show ?thesis using Ha' unfolding P_def by auto
 next
 assume Ha': a = 1
 with * show ?thesis unfolding r_def by auto
 qed
 qed

lemma r_bound:
 assumes Hn: $1 \leq n$
 shows $r(x/n) \leq H/T + (\text{if } n \in \text{region then } 2 + \ln(T/b) \text{ else } 0)$
proof (cases $n \in \text{region}$)
 assume *: $n \notin \text{region}$
 then consider $n < x - x/H \mid x + x/H < n$ unfolding region_def by auto
 hence $1 / |\ln(x/n)| \leq 2 * H$
proof cases
 have hH': $1 / (1 - 1/H) > 1$ using hH by auto
 case 1 hence $x/n > x / (x - x/H)$
 using Hx hH Hn by (intro divide_strict_left_mono) auto
 also have $x / (x - x/H) = 1 / (1 - 1/H)$
 using Hx hH by (auto simp add: field_simps)
 finally have xn: $x/n > 1 / (1 - 1/H)$.
 moreover have xn': $x/n > 1$ using xn hH' by linarith
 ultimately have $|\ln(x/n)| > \ln(1 / (1 - 1/H))$
 using hH Hx Hn by auto
 hence $1 / |\ln(x/n)| < 1 / \ln(1 / (1 - 1/H))$
 using xn' hH' by (intro divide_strict_left_mono mult_pos_pos ln_gt_zero) auto
 also have ... $\leq H$ **proof** -
 have $\ln(1 - 1/H) \leq -(1/H)$
 using hH by (intro ln_one_minus_pos_upper_bound) auto
 hence $-1 / \ln(1 - 1/H) \leq -1 / (-(1/H))$
 using hH by (intro divide_left_mono_neg) (auto intro: divide_neg_pos)
 also have ... = H by auto
 finally show ?thesis
 by (subst (2) inverse_eq_divide [symmetric])
 (subst ln_inverse, use hH in auto)
 qed
 finally show ?thesis using hH by auto
 next
 case 2 hence $x/n < x / (x + x/H)$
 using Hx hH Hn by (auto intro!: divide_strict_left_mono mult_pos_pos add_pos_pos)
 also have ... = $1 / (1 + 1/H)$
proof -
 have $0 < x + x * H$ using Hx hH by (auto intro: add_pos_pos)
 thus ?thesis using Hx hH by (auto simp add: field_simps)
 qed
 finally have xn: $x/n < 1 / (1 + 1/H)$.
 also have hH': ... < 1 using hH by (auto simp add: field_simps)
 finally have xn': $0 < x/n \wedge x/n < 1$ using Hx Hn by auto
 have $1 / |\ln(x/n)| = -1 / \ln(x/n)$
 using xn' by (auto simp add: field_simps)
 also have ... $\leq 2 * H$ **proof** -
 have $\ln(x/n) < \ln(1 / (1 + 1/H))$
 using xn xn' by (subst ln_less_cancel_iff) (blast, linarith)

also have $\dots = -\ln(1 + 1/H)$
by (*subst* (1) *inverse_eq_divide* [*symmetric*])
(subst *ln_inverse*, *intro* *add_pos_pos*, *use* *hH* **in** *auto*)
also have $\dots \leq -1/(2 * H)$
proof –
have $1/H - (1/H)^2 \leq \ln(1 + 1/H)$
by (*rule* *ln_one_plus_pos_lower_bound*) (*use* *hH* **in** *auto*)
hence $-\ln(1 + 1/H) \leq -1/H + (1/H)^2$ **by** *auto*
also have $\dots \leq -1/(2 * H)$
using *hH* **unfolding** *power2_eq_square* **by** (*auto simp add: field_simps*)
finally show *?thesis* .
qed
finally have $-1/\ln(x/n) \leq -1/(-1/(2 * H))$
by (*intro* *divide_left_mono_neg*) (*insert* *xn'* *hH*, *auto simp add: field_simps*)
thus *?thesis* **by** *auto*
qed
finally show *?thesis* .
qed
hence $(1/|\ln(x/n)|)/(2 * T) \leq (2 * H)/(2 * T)$
using *hT'* **by** (*intro* *divide_right_mono*) *auto*
hence $1/(2 * T * |\ln(x/n)|) \leq H/T$
by (*simp add: field_simps*)
moreover have $x/n \neq 1$ **using** ** hH* **unfolding** *region_def* **by** *auto*
ultimately show *?thesis* **unfolding** *r_def* **using** *** **by** *auto*
next
assume **: n* \in *region*
moreover have $2 + \ln(T/b) \leq H/T + (2 + \ln(T/b))$
using *hH hT'* **by** *auto*
ultimately show *?thesis* **unfolding** *r_def* **by** *auto*
qed

lemma *perron_aux*:
assumes *Hn*: $0 < n$
shows $\|F' n\| \leq 1/n \text{ nat_powr } b * (x \text{ powr } b * H/T)$
 $+ (\text{if } n \in \text{region then } 3 * (2 + \ln(T/b)) \text{ else } 0)$ (**is** *?P* \leq *?Q*)
proof –
have $\|F(x/n)\| \leq (x/n) \text{ powr } b * r(x/n)$
by (*rule* *perron_aux'*) (*use* *Hx Hn* **in** *auto*)
also have $\dots \leq (x/n) \text{ powr } b * (H/T + (\text{if } n \in \text{region then } 2 + \ln(T/b) \text{ else } 0))$
by (*intro* *mult_left_mono* *r_bound*) (*use* *Hn* **in** *auto*)
also have $\dots \leq ?Q$
proof –
have **:* $(x/n) \text{ powr } b * (H/T) = 1/n \text{ nat_powr } b * (x \text{ powr } b * H/T)$
using *Hx Hn* **by** (*subst* *powr_divide*) (*auto simp add: field_simps*)
moreover have $(x/n) \text{ powr } b * (H/T + (2 + \ln(T/b)))$
 $\leq 1/n \text{ nat_powr } b * (x \text{ powr } b * H/T) + 3 * (2 + \ln(T/b))$
when *Hn'*: $n \in \text{region}$
proof –
have $(x/n) \text{ powr } b \leq 3$
proof –
have $x - x/H \leq n$ **using** *Hn'* **unfolding** *region_def* **by** *auto*
moreover have $x/H < x/1$ **using** *hH Hx* **by** (*intro* *divide_strict_left_mono*) *auto*
ultimately have $x/n \leq x/(x - x/H)$
using *Hx hH Hn* **by** (*intro* *divide_left_mono* *mult_pos_pos*) *auto*
also have $\dots = 1 + 1/(H - 1)$

```

    using Hx hH by (auto simp add: field_simps)
  finally have (x / n) powr b ≤ (1 + 1 / (H - 1)) powr b
    using Hx Hn Hb by (intro powr_mono2) auto
  also have ... ≤ exp (b / (H - 1))
proof -
  have ln (1 + 1 / (H - 1)) ≤ 1 / (H - 1)
    using hH by (intro ln_add_one_self_le_self) auto
  hence b * ln (1 + 1 / (H - 1)) ≤ b * (1 / (H - 1))
    using Hb by (intro mult_left_mono) auto
  thus ?thesis unfolding powr_def by auto
qed
also have ... ≤ exp 1 using Hb hH' by auto
also have ... ≤ 3 by (rule exp_le)
finally show ?thesis .
qed
moreover have 0 ≤ ln (T / b) using hT Hb by (auto intro!: ln_ge_zero)
ultimately show ?thesis using hT
  by (subst ring_distrib, subst *, subst add_le_cancel_left)
  (intro mult_right_mono, auto intro!: add_nonneg_nonneg)
qed
ultimately show ?thesis by auto
qed
finally show ?thesis unfolding F'_def .
qed

definition a where a n ≡ fds_nth f n

lemma finite_region: finite region
  unfolding region_def by (subst nat_le_real_iff) auto

lemma zero_notin_region: 0 ∉ region
  unfolding region_def using hH Hx by (auto simp add: field_simps)

lemma path_image_conv:
  assumes s ∈ img_path
  shows conv_abscissa f < s · 1
proof -
  from assms have Re s = b
    unfolding img_path_def path_def
  by (auto simp add: closed_segment_def legacy_Complex_simps field_simps)
  thus ?thesis using Hb' conv_le_abs_conv_abscissa [of f] by auto
qed

lemma converge_on_path:
  assumes s ∈ img_path
  shows fds_converges f s
  by (intro fds_converges path_image_conv assms)

lemma summable_on_path:
  assumes s ∈ img_path
  shows (λn. a n / n nat_powr s) subsummable {1..}
  unfolding a_def by (intro eval_fds_complex_subsum(2) converge_on_path assms)

lemma zero_notin_path:
  shows 0 ∉ closed_segment (Complex b (- T)) (Complex b T)

```

using *Hb unfolding img_path_def path_def*
 by (auto simp add: closed_segment_def legacy_Complex_simps field_simps)

lemma *perron_bound*:

$$\|\sum 'n \geq 1. a n * F' n\| \leq x \text{ powr } b * H * B / T$$

$$+ 3 * (2 + \ln (T / b)) * (\sum n \in \text{region}. \|a n\|)$$

proof –

define *M* where $M \equiv 3 * (2 + \ln (T / b))$

have *sum_1*: $(\lambda n. \|a n / n \text{ nat_powr } (b :: \text{complex})\|)$ subsummable $\{1..\}$

unfolding *a_def*

by (fold *nat_power_complex_def*)

(fastforce intro: *Hb' fds_abs_subsummable fds_abs_converges*)

hence *sum_2*: $(\lambda n. \|a n\| * 1 / n \text{ nat_powr } b)$ subsummable $\{1..\}$

proof –

have $\|a n / n \text{ nat_powr } (b :: \text{complex})\| = \|a n\| * 1 / n \text{ nat_powr } b$ for *n*

by (auto simp add: norm_divide field_simps norm_powr_real_powr')

thus ?thesis using *sum_1* by auto

qed

hence *sum_3*: $(\lambda n. \|a n\| * 1 / n \text{ nat_powr } b * (x \text{ powr } b * H / T))$ subsummable $\{1..\}$

by (rule subsummable_mult2)

moreover have *sum_4*: $(\lambda n. \text{if } n \in \text{region} \text{ then } M * \|a n\| \text{ else } 0)$ subsummable $\{1..\}$

by (intro has_subsum_summable, rule has_subsum_If_finite)

(insert *finite_region*, auto)

moreover have $\|a n * F' n\|$

$$\leq \|a n\| * 1 / n \text{ nat_powr } b * (x \text{ powr } b * H / T)$$

$$+ (\text{if } n \in \text{region} \text{ then } M * \|a n\| \text{ else } 0) \text{ (is } ?x' \leq ?x)$$

when $n \in \{1..\}$ for *n*

proof –

$$\text{have } \|a n * F' n\| \leq \|a n\| *$$

$$(1 / n \text{ nat_powr } b * (x \text{ powr } b * H / T) + (\text{if } n \in \text{region} \text{ then } M \text{ else } 0))$$

unfolding *M_def*

by (subst norm_mult)

(intro mult_left_mono *perron_aux*, use that in auto)

also have ... = ?x by (simp add: field_simps)

finally show ?thesis .

qed

ultimately have $\|\sum 'n \geq 1. a n * F' n\|$

$$\leq (\sum 'n \geq 1. \|a n\| * 1 / n \text{ nat_powr } b * (x \text{ powr } b * H / T))$$

$$+ (\text{if } n \in \text{region} \text{ then } M * \|a n\| \text{ else } 0))$$

by (intro subsum_norm_bound subsummable_add)

also have ... $\leq x \text{ powr } b * H * B / T + M * (\sum n \in \text{region}. \|a n\|)$

proof –

$$\text{have } (\sum 'n \geq 1. (\text{if } n \in \text{region} \text{ then } M * \|a n\| \text{ else } 0))$$

$$= (\sum n \in \text{region} \cap \{1..\}. M * \|a n\|)$$

by (intro subsumI [symmetric] has_subsum_If_finite_set finite_region)

also have ... = $M * (\sum n \in \text{region}. \|a n\|)$

proof –

$$\text{have } \text{region} \cap \{1..\} = \text{region}$$

using zero_notin_region zero_less_iff_neq_zero by (auto intro: Suc_leI)

thus ?thesis by (subst sum_distrib_left) (use zero_notin_region in auto)

qed

also have

$$(\sum 'n \geq 1. \|a n\| * 1 / n \text{ nat_powr } b * (x \text{ powr } b * H / T))$$

$$\leq x \text{ powr } b * H * B / T$$

by (subst subsum_mult2, rule sum_2, insert hB hH hT', fold a_def)

```

      (auto simp add: field_simps, subst (1) mult.commute, auto intro: mult_right_mono)
    ultimately show ?thesis
      by (subst subsum_add [symmetric]) ((rule sum_3 sum_4)+, auto)
  qed
  finally show ?thesis unfolding M_def .
qed

```

lemma perron:

```

  (λs. eval_fds f s * x powr s / s) contour_integrable_on path
  ||sum_upto a x - 1 / (2 * pi * i) * contour_integral path (λs. eval_fds f s * x powr s / s)||
  ≤ x powr b * H * B / T + 3 * (2 + ln (T / b)) * (∑ n∈region. ||a n||)
proof (goal_cases)
  define g where g s ≡ eval_fds f s * x powr s / s for s :: complex
  define h where h s n ≡ a n / n nat_powr s * (x powr s / s) for s :: complex and n :: nat
  define G where G n ≡ contour_integral path (λs. (x / n) powr s / s) for n :: nat
  define H where H n ≡ 1 / (2 * pi * i) * G n for n :: nat
  have h_integrable: (λs. h s n) contour_integrable_on path when 0 < n for n
    using Hb Hx unfolding path_def h_def
    by (intro contour_integrable_continuous_linepath continuous_intros)
      (use that zero_notin_path in auto)
  have contour_integral_path g = contour_integral_path (λs. ∑ 'n ≥ 1. h s n)
  proof (rule contour_integral_eq, fold img_path_def)
    fix s assume *: s ∈ img_path
    hence g s = (∑ 'n ≥ 1. a n / n nat_powr s) * (x powr s / s)
      unfolding g_def a_def
      by (subst eval_fds_complex_subsum) (auto intro!: converge_on_path)
    also have ... = (∑ 'n ≥ 1. a n / n nat_powr s * (x powr s / s))
      by (intro subsum_mult2 [symmetric] summable) (intro summable_on_path *)
    finally show g s = (∑ 'n ≥ 1. h s n) unfolding h_def .
  qed
  also have
    sum_1: (λn. contour_integral_path (λs. h s n)) subsummable {1..}
    and ... = (∑ 'n ≥ 1. contour_integral_path (λs. h s n))
  proof (goal_cases)
    have ((λN. contour_integral_path (λs. sum (h s) {1..N}))
      → contour_integral_path (λs. subsum (h s) {1..})) at_top
    proof (rule contour_integral_uniform_limit)
      show valid_path path unfolding path_def by auto
      show sequentially ≠ bot by auto
    next
      fix t :: real
      show ||vector_derivative path (at t)|| ≤ sqrt (4 * T2)
        unfolding path_def by (auto simp add: legacy_Complex_simps)
    next
      from path_image_conv
      have *: uniformly_convergent_on img_path (λN s. ∑ n≤N. fds_nth f n / nat_power n s)
        by (intro uniformly_convergent_eval_fds) (unfold path_def img_path_def, auto)
      have *: uniformly_convergent_on img_path (λN s. ∑ n = 1..N. a n / n nat_powr s)
      proof -
        have (∑ n≤N. fds_nth f n / nat_power n s) = (∑ n = 1..N. a n / n nat_powr s) for N s
      proof -
        have (∑ n≤N. fds_nth f n / nat_power n s) = (∑ n≤N. a n / n nat_powr s)
          unfolding a_def nat_power_complex_def by auto
        also have ... = (∑ n∈{..N} - {0}. a n / n nat_powr s)
          by (subst sum_diff1) auto
      qed
    qed
  qed

```



```

also have ... = ( $\sum n = 1..N. a n / n \text{ nat\_powr } s$ )
proof -
  have  $\{..N\} - \{0\} = \{1..N\}$  by auto
  thus ?thesis by auto
qed
finally show ?thesis by auto
qed
thus ?thesis using * by auto
qed
hence uniform_limit img_path
( $\lambda N s. \sum n = 1..N. a n / n \text{ nat\_powr } s$ )
( $\lambda s. \sum 'n \geq 1. a n / n \text{ nat\_powr } s$ ) at_top
proof -
  have uniform_limit img_path
( $\lambda N s. \sum n = 1..N. a n / n \text{ nat\_powr } s$ )
( $\lambda s. \text{lim } (\lambda N. \sum n = 1..N. a n / n \text{ nat\_powr } s)$ ) at_top
  using * by (subst (asm) uniformly_convergent_uniform_limit_iff)
moreover have  $\text{lim } (\lambda N. \sum n = 1..N. a n / n \text{ nat\_powr } s) = (\sum 'n \geq 1. a n / n \text{ nat\_powr } s)$  for
s
  by (rule subsum_ge_limit)
ultimately show ?thesis by auto
qed
moreover have bounded (( $\lambda s. \text{subsum } (\lambda n. a n / n \text{ nat\_powr } s) \{1..\}$ ) 'img_path) (is bounded ?A)
proof -
  have bounded (eval_fds f 'img_path)
  by (intro compact_imp_bounded compact_continuous_image continuous_on_eval_fds)
  (use path_image_conv img_path_def path_def in auto)
moreover have ... = ?A
  unfolding a_def by (intro image_cong refl eval_fds_complex_subsum(1) converge_on_path)
ultimately show ?thesis by auto
qed
moreover have  $0 \notin \text{closed\_segment } (\text{Complex } b (- T)) (\text{Complex } b T)$ 
  using Hb by (auto simp: closed_segment_def legacy_Complex_simps algebra_simps)
hence bounded (( $\lambda s. x \text{ powr } s / s$ ) 'img_path)
  unfolding img_path_def path_def using Hx Hb
  by (intro compact_imp_bounded compact_continuous_image continuous_intros) auto
ultimately have uniform_limit img_path
( $\lambda N s. (\sum n = 1..N. a n / n \text{ nat\_powr } s) * (x \text{ powr } s / s)$ )
( $\lambda s. (\sum 'n \geq 1. a n / n \text{ nat\_powr } s) * (x \text{ powr } s / s)$ ) at_top (is ?P)
  by (intro uniform_lim_mult uniform_limit_const)
moreover have ?P = uniform_limit (path_image path)
( $\lambda N s. \text{sum } (h s) \{1..N\}$ ) ( $\lambda s. \text{subsum } (h s) \{1..\}$ ) at_top (is ?P = ?Q)
  unfolding h_def
  by (fold img_path_def, rule uniform_limit_cong', subst sum_distrib_right [symmetric], rule refl)
  (subst subsum_mult2, intro summable_on_path, auto)
ultimately show ?Q by blast
next
from h_integrable
show  $\forall_F N$  in at_top. ( $\lambda s. \text{sum } (h s) \{1..N\}$ ) contour_integrable_on path
  unfolding h_def by (intro eventuallyI contour_integrable_sum) auto
qed
hence *: has_subsum ( $\lambda n. \text{contour\_integral path } (\lambda s. h s n)$ )  $\{1..\}$  (contour_integral path ( $\lambda s. \text{subsum } (h s) \{1..\}$ ))
  using h_integrable by (subst (asm) contour_integral_sum) (auto intro: has_subsum_ge_limit)
case 1 from * show ?case unfolding h_def by (intro has_subsum_summable)

```

case 2 from * show ?case unfolding h_def by (rule subsumI)
qed
note this(2) also have
*sum_2: ($\lambda n. a n * G n$) subsummable {1..}*
and ... = ($\sum 'n \geq 1. a n * G n$)
proof (goal_cases)
have *: $a n * G n = \text{contour_integral path } (\lambda s. h s n)$ when $Hn: n \in \{1..\}$ for $n :: nat$
proof -
have ($\lambda s. (x / n) \text{ powr } s / s$) contour_integrable_on path
unfolding path_def by (rule perron_integrable) (use $Hn Hx hT$ in auto)
moreover have contour_integral path ($\lambda s. h s n$) = contour_integral path ($\lambda s. a n * ((x / n) \text{ powr } s / s)$)
proof (intro contour_integral_cong refl)
fix $s :: complex$
have ($(x / n) \text{ powr } s * n \text{ powr } s = ((x / n :: complex) * n) \text{ powr } s$
by (rule powr_times_real [symmetric]) (use $Hn Hx$ in auto)
also have ... = $x \text{ powr } s$ using Hn by auto
finally have ($(x / n) \text{ powr } s = x \text{ powr } s / n \text{ powr } s$ using Hn by (intro eq_divide_imp) auto
thus $h s n = a n * ((x / n) \text{ powr } s / s)$ unfolding h_def by (auto simp add: field_simps)
qed
ultimately show ?thesis unfolding G_def by (subst (asm) contour_integral_lmul) auto
qed
case 1 show ?case by (subst subsummable_cong) (use * sum_1 in auto)
case 2 show ?case by (intro subsum_cong * [symmetric])
qed
note this(2) finally have
 $1 / (2 * pi * i) * \text{contour_integral path } g = (\sum 'n \geq 1. a n * G n) * (1 / (2 * pi * i))$ **by auto**
also have
*sum_3: ($\lambda n. a n * G n * (1 / (2 * pi * i))$) subsummable {1..}*
and ... = ($\sum 'n \geq 1. a n * G n * (1 / (2 * pi * i))$)
by (intro subsummable_mult2 subsum_mult2 [symmetric] sum_2)+
note this(2) also have
*sum_4: ($\lambda n. a n * H n$) subsummable {1..}*
and ... = ($\sum 'n \geq 1. a n * H n$)
unfolding H_def using sum_3 by auto
note this(2) also have
 $\dots - (\sum 'n \geq 1. \text{if } n \leq x \text{ then } a n \text{ else } 0)$
 $= (\sum 'n \geq 1. a n * H n - (\text{if } n \leq x \text{ then } a n \text{ else } 0))$
using sum_4
by (rule subsum_minus(1), unfold subsummable_def)
(auto simp add: if_if_eq_conj nat_le_real_iff)
moreover have ($\sum 'n \geq 1. \text{if } n \leq x \text{ then } a n \text{ else } 0$) = sum_upto a x
proof -
have ($\sum 'n \geq 1. \text{if } n \leq x \text{ then } a n \text{ else } 0$) = ($\sum n :: nat | n \in \{1..\} \wedge n \leq x. a n$)
by (intro subsumI [symmetric] has_subsum_If_finite) (auto simp add: nat_le_real_iff)
also have ... = sum_upto a x
proof -
have $\{n :: nat. n \in \{1..\} \wedge n \leq x\} = \{n. 0 < n \wedge n \leq x\}$ by auto
thus ?thesis unfolding sum_upto_def by auto
qed
finally show ?thesis .
qed
moreover have ($\sum 'n \geq 1. a n * H n - (\text{if } n \leq x \text{ then } a n \text{ else } 0)$) = ($\sum 'n \geq 1. a n * F' n$)
unfolding F_def F'_def G_def H_def by (rule subsum_cong) (auto simp add: algebra_simps)
ultimately have result: $\| \text{sum_upto a x} - 1 / (2 * pi * i) * \text{contour_integral path } g \| = \| \sum 'n \geq 1. a$

```

n * F' n||
  by (subst norm_minus_commute) auto
case 1 show ?case
proof -
  have closed_segment (Complex b (- T)) (Complex b T)  $\subseteq$  {s. conv_abscissa f < ereal (s * 1)}
  using path_image_conv unfolding img_path_def path_def by auto
  thus ?thesis unfolding path_def
    by (intro contour_integrable_continuous_linepath continuous_intros)
      (use Hx zero_notin_path in auto)
qed
case 2 show ?case using perron_bound result unfolding g_def by linarith
qed
end

theorem perron_formula:
  fixes b B H T x :: real and f :: complex fds
  assumes Hb: 0 < b and hT: b  $\leq$  T
    and Hb': abs_conv_abscissa f < b
    and hH: 2  $\leq$  H and hH': b + 1  $\leq$  H and Hx: 2  $\leq$  x
    and hB: ( $\sum$  'n  $\geq$  1. ||fds_nth f n|| / n nat_powr b)  $\leq$  B
  shows ( $\lambda$ s. eval_fds f s * x powr s / s) contour_integrable_on (linepath (Complex b (-T)) (Complex b T))
    ( $\|$ sum_upto (fds_nth f) x - 1 / (2 * pi * i) *
      contour_integral (linepath (Complex b (-T)) (Complex b T)) ( $\lambda$ s. eval_fds f s * x powr s / s)||
       $\leq$  x powr b * H * B / T + 3 * (2 + ln (T / b)) * ( $\sum$  n | x - x / H  $\leq$  n  $\wedge$  n  $\leq$  x + x / H.
 $\|$ fds_nth f n||)
proof (goal_cases)
  interpret z: perron_locale using assms unfolding perron_locale_def by auto
  case 1 show ?case using z.perron(1) unfolding z.path_def .
  case 2 show ?case using z.perron(2) unfolding z.path_def z.region_def z.a_def .
qed

theorem perron_asymp:
  fixes b x :: real
  assumes b: b > 0 ereal b > abs_conv_abscissa f
  assumes x: x  $\geq$  2 x  $\notin$   $\mathbb{N}$ 
  defines L  $\equiv$  ( $\lambda$ T. linepath (Complex b (-T)) (Complex b T))
  shows (( $\lambda$ T. contour_integral (L T) ( $\lambda$ s. eval_fds f s * of_real x powr s / s))
     $\longrightarrow$  2 * pi * i * sum_upto ( $\lambda$ n. fds_nth f n) x) at_top
proof -
  define R where R = ( $\lambda$ H. {n. x - x / H  $\leq$  real n  $\wedge$  real n  $\leq$  x + x / H})
  have R_altdef: R H = {n. dist (of_nat n) x  $\leq$  x / H} for H
    unfolding R_def by (intro Collect_cong) (auto simp: dist_norm)
  obtain H where H: H  $\geq$  2 H  $\geq$  b + 1 R H = (if x  $\in$   $\mathbb{N}$  then {nat [x]} else {})
  proof (cases x  $\in$   $\mathbb{N}$ )
    case True
    then obtain m where [simp]: x = of_nat m by (elim Nats_cases)
    define H where H = Max {2, b + 1, x / 2}
    have H: H  $\geq$  2 H  $\geq$  b + 1 H  $\geq$  x / 2
      unfolding H_def by (rule Max.coboundedI; simp)+
    show ?thesis
  proof (rule that[of H])
    have n  $\notin$  R H if n  $\neq$  m for n :: nat
    proof -
      have x / H  $\leq$  x / (x / 2)

```

```

    by (intro divide_left_mono) (use H x in auto)
  hence  $x / H < 1$  using x by simp
  also have ...  $\leq |int\ n - int\ m|$  using  $\langle n \neq m \rangle$  by linarith
  also have ... = dist (of_nat n) x
    unfolding  $\langle x = of\_nat\ m \rangle$  dist_of_nat by simp
  finally show  $n \notin R\ H$  by (simp add: R_altdef)
qed
moreover have  $m \in R\ H$  using x by (auto simp: R_def)
ultimately show  $R\ H = (if\ x \in \mathbb{N}\ then\ \{nat\ \lfloor x \rfloor\}\ else\ \{\})$  by auto
qed (use H in auto)
next
case False
define d where  $d = setdist\ \{x\}\ \mathbb{N}$ 
have  $0 \in (\mathbb{N} :: real\ set)$  by auto
hence  $(\mathbb{N} :: real\ set) \neq \{\}$  by blast
hence  $d > 0$ 
  unfolding d_def using False by (subst setdist_gt_0_compact_closed) auto
define H where  $H = Max\ \{2, b + 1, 2 * x / d\}$ 
have H:  $H \geq 2\ H \geq b + 1\ H \geq 2 * x / d$ 
  unfolding H_def by (rule Max.coboundedI; simp)+

show ?thesis
proof (rule that[of H])
  have  $n \notin R\ H$  for  $n :: nat$ 
  proof -
    have  $x / H \leq x / (2 * x / d)$ 
      using H x  $\langle d > 0 \rangle$ 
      by (intro divide_left_mono) (auto intro!: mult_pos_pos)
    also have ...  $< d$ 
      using x  $\langle d > 0 \rangle$  by simp
    also have  $d \leq dist\ (of\_nat\ n)\ x$ 
      unfolding d_def by (subst dist_commute, rule setdist_le_dist) auto
    finally show  $n \notin R\ H$ 
      by (auto simp: R_altdef)
  qed
  thus  $R\ H = (if\ x \in \mathbb{N}\ then\ \{nat\ \lfloor x \rfloor\}\ else\ \{\})$ 
    using False by auto
qed (use H in auto)
qed

define g where  $g = (\lambda s. eval\_fds\ f\ s * of\_real\ x\ powr\ s / s)$ 
define I where  $I = (\lambda T. contour\_integral\ (L\ T)\ g)$ 
define c where  $c = 2 * pi * i$ 
define A where  $A = sum\_upto\ (fds\_nth\ f)$ 
define B where  $B = subsum\ (\lambda n. norm\ (fds\_nth\ f\ n) / n\ nat\_powr\ b)\ \{0+\dots\}$ 
define X where  $X = (if\ x \in \mathbb{Z}\ then\ \{nat\ \lfloor x \rfloor\}\ else\ \{\})$ 

have norm_le:  $norm\ (A\ x - I\ T / c) \leq x\ powr\ b * H * B / T$  if  $T: T \geq b$  for T
proof -
  interpret perron_locale b B H T x f
    by standard (use b T x H(1,2) in  $\langle auto\ simp: B\_def \rangle$ )
  from perron
  have  $norm\ (A\ x - I\ T / c) \leq x\ powr\ b * H * B / T$ 
    +  $3 * (\sum_{n \in R\ H} norm\ (fds\_nth\ f\ n)) * (2 + \ln\ (T / b))$ 
    by (simp add: I_def A_def g_def a_def local.path_def L_def c_def R_def)

```

```

      region_def algebra_simps)
    also have  $(\sum_{n \in R} H. \text{norm } (f \text{ds\_nth } f \ n)) = 0$ 
      using  $x \ H$  by auto
    finally show  $\text{norm } (A \ x - I \ T / c) \leq x \ \text{powr } b * H * B / T$ 
      by simp
  qed
  have eventually  $(\lambda T. \text{norm } (A \ x - I \ T / c) \leq x \ \text{powr } b * H * B / T)$  at_top
    using eventually_ge_at_top[of b] by eventually_elim (use norm_le in auto)
  moreover have  $(\lambda T. x \ \text{powr } b * H * B / T) \longrightarrow 0$  at_top
    by real_asymp
  ultimately have lim:  $(\lambda T. A \ x - I \ T / c) \longrightarrow 0$  at_top
    using Lim_null_comparison by fast
  have  $(\lambda T. -c * (A \ x - I \ T / c) + c * A \ x) \longrightarrow -c * 0 + c * A \ x$  at_top
    by (rule tendsto_intros lim)+
  also have  $(\lambda T. -c * (A \ x - I \ T / c) + c * A \ x) = I$ 
    by (simp add: algebra_simps c_def)
  finally show ?thesis
    by (simp add: c_def A_def I_def g_def)
  qed

```

```

unbundle no_pnt_notation
end
theory PNT_with_Remainder
imports
  Relation_of_PNTs
  Zeta_Zerofree
  Perron_Formula
begin
unbundle pnt_notation

```

6 Estimation of the order of $\frac{\zeta'(s)}{\zeta(s)}$

```

notation primes_psi ( $\psi$ )

```

```

lemma zeta_div_bound':

```

```

  assumes  $1 + \exp(-4 * \ln(14 + 4 * t)) \leq \sigma$ 
    and  $13 / 22 \leq t$ 
    and  $z \in \text{cball } (\text{Complex } \sigma \ t) \ (1 / 2)$ 
  shows  $\|\text{zeta } z / \text{zeta } (\text{Complex } \sigma \ t)\| \leq \exp(12 * \ln(14 + 4 * t))$ 

```

```

proof -

```

```

  interpret z: zeta_bound_param_2
     $\lambda t. 1 / 2 \ \lambda t. 4 * \ln(12 + 2 * \max 0 \ t) \ t \ \sigma \ t$ 
  unfolding zeta_bound_param_1_def zeta_bound_param_2_def
    zeta_bound_param_1_axioms_def zeta_bound_param_2_axioms_def
  using assms by (auto intro: classical_zeta_bound.zeta_bound_param_axioms)
  show ?thesis using z.zeta_div_bound assms(2) assms(3)
    unfolding z.s_def z.r_def by auto

```

```

qed

```

```

lemma zeta_div_bound:

```

```

  assumes  $1 + \exp(-4 * \ln(14 + 4 * |t|)) \leq \sigma$ 
    and  $13 / 22 \leq |t|$ 
    and  $z \in \text{cball } (\text{Complex } \sigma \ t) \ (1 / 2)$ 
  shows  $\|\text{zeta } z / \text{zeta } (\text{Complex } \sigma \ t)\| \leq \exp(12 * \ln(14 + 4 * |t|))$ 
  proof (cases  $0 \leq t$ )

```

```

case True with assms(2) have  $13 / 22 \leq t$  by auto
thus ?thesis using assms by (auto intro: zeta_div_bound')
next
case False with assms(2) have Ht: t ≤ - 13 / 22 by auto
moreover have  $1: \text{Complex } \sigma (- t) = \text{cnj } (\text{Complex } \sigma t)$  by (auto simp add: legacy_Complex_simps)
ultimately have  $\|\text{zeta } (\text{cnj } z) / \text{zeta } (\text{Complex } \sigma (- t))\| \leq \exp (12 * \ln (14 + 4 * (- t)))$ 
using assms(1) assms(3)
by (intro zeta_div_bound', auto simp add: dist_complex_def)
(subst complex_cnj_diff [symmetric], subst complex_mod_cnj)
thus ?thesis using Ht by (subst (asm) 1) (simp add: norm_divide)
qed

```

definition C_2 **where** $C_2 \equiv 319979520 :: \text{real}$

lemma $C_2_gt_zero: 0 < C_2$ **unfolding** C_2_def **by** *auto*

lemma *logderiv_zeta_order_estimate'*:

$\forall_F t$ *in* (*abs going_to at_top*).

$\forall \sigma. 1 - 1 / 7 * C_1 / \ln (|t| + 3) \leq \sigma$

$\longrightarrow \|\text{logderiv zeta } (\text{Complex } \sigma t)\| \leq C_2 * (\ln (|t| + 3))^2$

proof –

define F **where** $F :: \text{real filter} \equiv \text{abs going_to at_top}$

define r **where** $r t \equiv C_1 / \ln (|t| + 3)$ **for** $t :: \text{real}$

define s **where** $s \sigma t \equiv \text{Complex } (\sigma + 2 / 7 * r t) t$ **for** σt

have $r_nonneg: 0 \leq r t$ **for** t **unfolding** $PNT_const_C_1_def r_def$ **by** *auto*

have $\|\text{logderiv zeta } (\text{Complex } \sigma t)\| \leq C_2 * (\ln (|t| + 3))^2$

when $h: 1 - 1 / 7 * r t \leq \sigma$

$\exp (- 4 * \ln (14 + 4 * |t|)) \leq 1 / 7 * r t$

$8 / 7 * r t \leq |t|$

$8 / 7 * r t \leq 1 / 2$

$13 / 22 \leq |t|$ **for** σt

proof –

have $\|\text{logderiv zeta } (\text{Complex } \sigma t)\| \leq 8 * (12 * \ln (14 + 4 * |t|)) / (8 / 7 * r t)$

proof (*rule lemma_3_9_beta1' [where ?s = s σ t], goal_cases*)

case 1 **show** *?case* **unfolding** $PNT_const_C_1_def r_def$ **by** *auto*

case 2 **show** *?case* **by** *auto*

have $\text{notin_ball}: 1 \notin \text{ball } (s \sigma t) (8 / 7 * r t)$

proof –

note $h(3)$

also **have** $|t| = |\text{Im } (\text{Complex } (\sigma + 2 / 7 * r t) t - 1)|$ **by** *auto*

also **have** $\dots \leq \|\text{Complex } (\sigma + 2 / 7 * r t) t - 1\|$ **by** (*rule abs_Im_le_cmod*)

finally **show** *?thesis*

unfolding s_def **by** (*auto simp add: dist_complex_def*)

qed

case 3 **show** *?case* **by** (*intro holomorphic_zeta notin_ball*)

case 6 **show** *?case*

using r_nonneg **unfolding** s_def

by (*auto simp add: dist_complex_def legacy_Complex_simps*)

fix z **assume** $H z: z \in \text{ball } (s \sigma t) (8 / 7 * r t)$

show $\text{zeta } z \neq 0$

proof (*rule ccontr*)

assume $\neg \text{zeta } z \neq 0$

hence $\text{zero}: \text{zeta } (\text{Complex } (\text{Re } z) (\text{Im } z)) = 0$ **by** *auto*

have $r t \leq C_1 / \ln (|\text{Im } z| + 2)$

proof –

have $\|s \sigma t - z\| < 1$
using $H_z h(4)$ **by** $(auto \text{ simp add: dist_complex_def})$
hence $|t - \text{Im } z| < 1$
using $abs_Im_le_cmod$ $[of \ s \ \sigma \ t - z]$
unfolding s_def **by** $(auto \text{ simp add: legacy_Complex_simps})$
hence $|\text{Im } z| < |t| + 1$ **by** $auto$
thus $?thesis$ **unfolding** r_def
by $(intro \ divide_left_mono \ mult_pos_pos)$
 $(subst \ ln_le_cancel_iff, \ use \ C_1_gt_zero \ in \ auto)$
qed
also have $\dots \leq 1 - \text{Re } z$
using $notin_ball \ H_z$ **by** $(intro \ zeta_nonzero_region \ zero) \ auto$
also have $\dots < 1 - \text{Re } (s \ \sigma \ t) + 8 / 7 * r \ t$
proof –
have $\text{Re } (s \ \sigma \ t - z) \leq |\text{Re } (s \ \sigma \ t - z)|$ **by** $auto$
also have $\dots < 8 / 7 * r \ t$
using $H_z \ abs_Re_le_cmod$ $[of \ s \ \sigma \ t - z]$
by $(auto \text{ simp add: dist_complex_def})$
ultimately show $?thesis$ **by** $auto$
qed
also have $\dots = 1 - \sigma + 6 / 7 * r \ t$ **unfolding** s_def **by** $auto$
also have $\dots \leq r \ t$ **using** $h(1)$ **by** $auto$
finally show $False$ **by** $auto$
qed
from H_z **have** $z \in cball \ (s \ \sigma \ t) \ (1 / 2)$
using $h(4)$ **by** $auto$
thus $\|zeta \ z / zeta \ (s \ \sigma \ t)\| \leq exp \ (12 * ln \ (14 + 4 * |t|))$
using $h(1) \ h(2)$ **unfolding** s_def
by $(intro \ zeta_div_bound \ h(5)) \ auto$
qed
also have $\dots = 84 / r \ t * ln \ (14 + 4 * |t|)$
by $(auto \text{ simp add: field_simps})$
also have $\dots \leq 336 / C_1 * ln \ (|t| + 2) * ln \ (|t| + 3)$
proof –
have $84 / r \ t * ln \ (14 + 4 * |t|) \leq 84 / r \ t * (4 * ln \ (|t| + 2))$
using r_nonneg **by** $(intro \ mult_left_mono \ mult_right_mono \ ln_bound_1) \ auto$
thus $?thesis$ **unfolding** r_def **by** $(simp \ add: \ mult_ac)$
qed
also have $\dots \leq 336 / C_1 * (ln \ (|t| + 3))^2$
unfolding $power2_eq_square$
by $(simp \ add: \ mult_ac, \ intro \ divide_right_mono \ mult_right_mono)$
 $(subst \ ln_le_cancel_iff, \ use \ C_1_gt_zero \ in \ auto)$
also have $\dots = C_2 * (ln \ (|t| + 3))^2$
unfolding $PNT_const_C_1_def \ C_2_def$ **by** $auto$
finally show $?thesis$.
qed
hence
 $\forall_F \ t \ in \ F.$
 $exp \ (- \ 4 * ln \ (14 + 4 * |t|)) \leq 1 / 7 * r \ t$
 $\rightarrow 8 / 7 * r \ t \leq |t|$
 $\rightarrow 8 / 7 * r \ t \leq 1 / 2$
 $\rightarrow 13 / 22 \leq |t|$
 $\rightarrow (\forall \sigma. \ 1 - 1 / 7 * r \ t \leq \sigma$
 $\rightarrow \|\logderiv \ zeta \ (Complex \ \sigma \ t)\| \leq C_2 * (ln \ (|t| + 3))^2)$
by $(blast \ intro: \ eventuallyI)$

moreover have $\forall_F t \text{ in } F. \exp(-4 * \ln(14 + 4 * |t|)) \leq 1 / 7 * r t$
unfolding $F_def\ r_def\ PNT_const_C1_def$
by $(rule\ eventually_going_toI)\ real_asymp$
moreover have $\forall_F t \text{ in } F. 8 / 7 * r t \leq |t|$
unfolding $F_def\ r_def\ PNT_const_C1_def$
by $(rule\ eventually_going_toI)\ real_asymp$
moreover have $\forall_F t \text{ in } F. 8 / 7 * r t \leq 1 / 2$
unfolding $F_def\ r_def\ PNT_const_C1_def$
by $(rule\ eventually_going_toI)\ real_asymp$
moreover have $\forall_F t \text{ in } F. 13 / 22 \leq |t|$
unfolding F_def **by** $(rule\ eventually_going_toI)\ real_asymp$
ultimately have
 $\forall_F t \text{ in } F. (\forall \sigma. 1 - 1 / 7 * r t \leq \sigma$
 $\longrightarrow \|\logderiv\ zeta\ (Complex\ \sigma\ t)\| \leq C_2 * (\ln(|t| + 3))^2)$
by $eventually_elim\ blast$
thus $?thesis$ **unfolding** $F_def\ r_def$ **by** $auto$
qed

definition C_3 **where**

$C_3 \equiv SOME\ T. 0 < T \wedge$
 $(\forall t. T \leq |t| \longrightarrow$
 $(\forall \sigma. 1 - 1 / 7 * C_1 / \ln(|t| + 3) \leq \sigma$
 $\longrightarrow \|\logderiv\ zeta\ (Complex\ \sigma\ t)\| \leq C_2 * (\ln(|t| + 3))^2))$

lemma C_3_prop :

$0 < C_3 \wedge$
 $(\forall t. C_3 \leq |t| \longrightarrow$
 $(\forall \sigma. 1 - 1 / 7 * C_1 / \ln(|t| + 3) \leq \sigma$
 $\longrightarrow \|\logderiv\ zeta\ (Complex\ \sigma\ t)\| \leq C_2 * (\ln(|t| + 3))^2))$

proof –

obtain T' **where** hT :
 $\wedge t. T' \leq |t| \implies$
 $(\forall \sigma. 1 - 1 / 7 * C_1 / \ln(|t| + 3) \leq \sigma$
 $\longrightarrow \|\logderiv\ zeta\ (Complex\ \sigma\ t)\| \leq C_2 * (\ln(|t| + 3))^2)$
using $logderiv_zeta_order_estimate'$
 $[unfolded\ going_to_def, THEN\ rev_iffD1,$
 $OF\ eventually_filtercomap_at_top_linorder]$ **by** $blast$
define T **where** $T \equiv max\ 1\ T'$
show $?thesis$ **unfolding** C_3_def
by $(rule\ someI\ [of_ T])\ (unfold\ T_def, use\ hT\ in\ auto)$

qed

lemma $C_3_gt_zero$: $0 < C_3$ **using** C_3_prop **by** $blast$

lemma $logderiv_zeta_order_estimate$:

assumes $1 - 1 / 7 * C_1 / \ln(|t| + 3) \leq \sigma\ C_3 \leq |t|$
shows $\|\logderiv\ zeta\ (Complex\ \sigma\ t)\| \leq C_2 * (\ln(|t| + 3))^2$
using $assms\ C_3_prop$ **by** $blast$

definition $zeta_zerofree_region$

where $zeta_zerofree_region \equiv \{s. s \neq 1 \wedge 1 - C_1 / \ln(|Im\ s| + 2) < Re\ s\}$

definition $logderiv_zeta_region$

where $logderiv_zeta_region \equiv \{s. C_3 \leq |Im\ s| \wedge 1 - 1 / 7 * C_1 / \ln(|Im\ s| + 3) \leq Re\ s\}$

definition $zeta_strip_region$

where $zeta_strip_region\ \sigma\ T \equiv \{s. s \neq 1 \wedge \sigma \leq Re\ s \wedge |Im\ s| \leq T\}$

definition *zeta_strip_region'*

where $zeta_strip_region' \sigma T \equiv \{s. s \neq 1 \wedge \sigma \leq Re\ s \wedge C_3 \leq |Im\ s| \wedge |Im\ s| \leq T\}$

lemma *strip_in_zerofree_region:*

assumes $1 - C_1 / \ln(T + 2) < \sigma$

shows $zeta_strip_region' \sigma T \subseteq zeta_zerofree_region$

proof

fix s assume $Hs: s \in zeta_strip_region' \sigma T$

hence $Hs': s \neq 1 \ \sigma \leq Re\ s \ |Im\ s| \leq T$ **unfolding** *zeta_strip_region'_def* **by** *auto*

from *this(3)* **have** $C_1 / \ln(T + 2) \leq C_1 / \ln(|Im\ s| + 2)$

using *C1_gt_zero* **by** (*intro divide_left_mono mult_pos_pos*) *auto*

thus $s \in zeta_zerofree_region$ **using** Hs' *assms* **unfolding** *zeta_zerofree_region_def* **by** *auto*

qed

lemma *strip_in_logderiv_zeta_region:*

assumes $1 - 1 / 7 * C_1 / \ln(T + 3) \leq \sigma$

shows $zeta_strip_region' \sigma T \subseteq logderiv_zeta_region$

proof

fix s assume $Hs: s \in zeta_strip_region' \sigma T$

hence $Hs': s \neq 1 \ \sigma \leq Re\ s \ C_3 \leq |Im\ s| \ |Im\ s| \leq T$ **unfolding** *zeta_strip_region'_def* **by** *auto*

from *this(4)* **have** $C_1 / (7 * \ln(T + 3)) \leq C_1 / (7 * \ln(|Im\ s| + 3))$

using *C1_gt_zero* **by** (*intro divide_left_mono mult_pos_pos*) *auto*

thus $s \in logderiv_zeta_region$ **using** Hs' *assms* **unfolding** *logderiv_zeta_region_def* **by** *auto*

qed

lemma *strip_condition_imp:*

assumes $0 \leq T \ 1 - 1 / 7 * C_1 / \ln(T + 3) \leq \sigma$

shows $1 - C_1 / \ln(T + 2) < \sigma$

proof –

have $\ln(T + 2) \leq 7 * \ln(T + 2)$ **using** *assms(1)* **by** *auto*

also have $\dots < 7 * \ln(T + 3)$ **using** *assms(1)* **by** *auto*

finally have $C_1 / (7 * \ln(T + 3)) < C_1 / \ln(T + 2)$

using *C1_gt_zero* *assms(1)* **by** (*intro divide_strict_left_mono mult_pos_pos*) *auto*

thus *?thesis* **using** *assms(2)* **by** *auto*

qed

lemma *zeta_zerofree_region:*

assumes $s \in zeta_zerofree_region$

shows $zeta\ s \neq 0$

using *zeta_nonzero_region* [*of Re s Im s*] *assms*

unfolding *zeta_zerofree_region_def* **by** *auto*

lemma *logderiv_zeta_region_estimate:*

assumes $s \in logderiv_zeta_region$

shows $\|logderiv\ zeta\ s\| \leq C_2 * (\ln(|Im\ s| + 3))^2$

using *logderiv_zeta_order_estimate* [*of Im s Re s*] *assms*

unfolding *logderiv_zeta_region_def* **by** *auto*

definition $C_4 :: real$ **where** $C_4 \equiv 1 / 6666241$

lemma *C4_prop:*

$\forall_F\ x\ in_at_top. C_4 / \ln\ x \leq C_1 / (7 * \ln(x + 3))$

unfolding *PNT_const_C1_def* *C4_def* **by** *real_asymp*

lemma *C4_gt_zero:* $0 < C_4$ **unfolding** *C4_def* **by** *auto*

definition C_5_prop **where**

C_5_prop $C_5 \equiv$
 $0 < C_5 \wedge (\forall_F x \text{ in } at_top. (\forall t. |t| \leq x$
 $\longrightarrow \|\logderiv\ zeta\ (Complex\ (1 - C_4 / \ln\ x)\ t)\| \leq C_5 * (\ln\ x)^2))$

lemma $logderiv_zeta_bound_vertical'$:

$\exists C_5. C_5_prop\ C_5$

proof –

define K **where** $K \equiv cbox\ (Complex\ 0\ (-C_3))\ (Complex\ 2\ C_3)$

define Γ **where** $\Gamma \equiv \{s \in K. zeta'\ s = 0\}$

have $zeta'\ not_zero_on\ K$

unfolding $not_zero_on_def\ K_def$ **using** $C_3_gt_zero$

by $(intro\ bexI\ [where\ x = 2])$

$(auto\ simp\ add: zeta_eq_zero_iff_zeta'\ zeta_2\ in_cbox_complex_iff)$

hence $fin: finite\ \Gamma$

unfolding $\Gamma_def\ K_def$

by $(auto\ intro!: convex_connected\ analytic_compact_finite_zeros\ zeta'_analytic)$

define α **where** $\alpha \equiv if\ \Gamma = \{\} \text{ then } 0 \text{ else } (1 + Max\ (Re\ ' \Gamma)) / 2$

define K' **where** $K' \equiv cbox\ (Complex\ \alpha\ (-C_3))\ (Complex\ 1\ C_3)$

have $H\alpha: \alpha \in \{0..<1\}$

proof $(cases\ \Gamma = \{\})$

case $True$ **thus** $?thesis$ **unfolding** α_def **by** $auto$

next

case $False$ **hence** $h\Gamma: \Gamma \neq \{\}$.

moreover **have** $Re\ a < 1$ **if** $Ha: a \in \Gamma$ **for** a

proof $(rule\ ccontr)$

assume $\neg Re\ a < 1$ **hence** $1 \leq Re\ a$ **by** $auto$

hence $zeta'\ a \neq 0$ **by** $(subst\ zeta'_eq_zero_iff)$ $(use\ zeta_Re_ge_1_nonzero\ in\ auto)$

thus $False$ **using** Ha **unfolding** Γ_def **by** $auto$

qed

moreover **have** $\exists a \in \Gamma. 0 \leq Re\ a$

proof –

from $h\Gamma$ **have** $\exists a. a \in \Gamma$ **by** $auto$

moreover **have** $\bigwedge a. a \in \Gamma \implies 0 \leq Re\ a$

unfolding $\Gamma_def\ K_def$ **by** $(auto\ simp\ add: in_cbox_complex_iff)$

ultimately **show** $?thesis$ **by** $auto$

qed

ultimately **have** $0 \leq Max\ (Re\ ' \Gamma)\ Max\ (Re\ ' \Gamma) < 1$

using fin **by** $(auto\ simp\ add: Max_ge_iff)$

thus $?thesis$ **unfolding** α_def **using** $h\Gamma$ **by** $auto$

qed

have $nonzero: zeta'\ z \neq 0$ **when** $z \in K'$ **for** z

proof $(rule\ ccontr)$

assume $\neg zeta'\ z \neq 0$

moreover **have** $K' \subseteq K$ **unfolding** $K'_def\ K_def$

by $(rule\ subset_box_imp)$ $(insert\ H\alpha, simp\ add: Basis_complex_def)$

ultimately **have** $Hz: z \in \Gamma$ **unfolding** Γ_def **using** $that$ **by** $auto$

hence $Re\ z \leq Max\ (Re\ ' \Gamma)$ **using** fin **by** $(intro\ Max_ge)\ auto$

also **have** $\dots < \alpha$

proof –

from Hz **have** $\Gamma \neq \{\}$ **by** $auto$

thus $?thesis$ **using** $H\alpha$ **unfolding** α_def **by** $auto$

qed

finally **have** $Re\ z < \alpha$.

moreover from $\langle z \in K' \rangle$ **have** $\alpha \leq \text{Re } z$
unfolding K'_def **by** (*simp add: in_cbox_complex_iff*)
ultimately show *False* **by** *auto*
qed
hence *logderiv zeta'* **analytic_on** K' **by** (*intro analytic_intros*)
moreover have *compact* K' **unfolding** K'_def **by** *auto*
ultimately have *bounded* (*logderiv zeta'*) ' K' '
by (*intro analytic_imp_holomorphic holomorphic_on_imp_continuous_on*
compact_imp_bounded compact_continuous_image)
from this [*THEN rev_iffD1, OF bounded_pos*]
obtain M **where**
 $hM: \bigwedge s. s \in K' \implies \|\text{logderiv zeta}' s\| \leq M$ **by** *auto*
have $(\lambda t. C_2 * (\ln(t + 3))^2) \in O(\lambda x. (\ln x)^2)$ **using** $C_2_gt_zero$ **by** *real_asymp*
then obtain γ **where**
 $H\gamma: \forall_F x \text{ in } \text{at_top}. \|C_2 * (\ln(x + 3))^2\| \leq \gamma * \|(\ln x)^2\|$
unfolding *bigo_def* **by** *auto*
define C_5 **where** $C_5 \equiv \max 1 \ \gamma$
have $C_5_gt_zero: 0 < C_5$ **unfolding** C_5_def **by** *auto*
have $\forall_F x \text{ in } \text{at_top}. \gamma * (\ln x)^2 \leq C_5 * (\ln x)^2$
by (*intro eventuallyI mult_right_mono*) (*unfold C5_def, auto*)
with $H\gamma$ **have** $hC_5: \forall_F x \text{ in } \text{at_top}. C_2 * (\ln(x + 3))^2 \leq C_5 * (\ln x)^2$
by *eventually_elim* (*use C2_gt_zero in auto*)
have $\|\text{logderiv zeta}(\text{Complex}(1 - C_4 / \ln x) t)\| \leq C_5 * (\ln x)^2$
when $h: C_3 \leq |t| \ |t| \leq x \ 1 < x$
 $C_4 / \ln x \leq C_1 / (7 * \ln(x + 3))$
 $C_2 * (\ln(x + 3))^2 \leq C_5 * (\ln x)^2$ **for** $x \ t$
proof –
have $\text{Re}(\text{Complex}(1 - C_4 / \ln x) t) \neq \text{Re } 1$ **using** $C_4_gt_zero$ $h(3)$ **by** *auto*
hence $\text{Complex}(1 - C_4 / \ln x) t \neq 1$ **by** *metis*
hence $\text{Complex}(1 - C_4 / \ln x) t \in \text{zeta_strip_region}'(1 - C_4 / \ln x) x$
unfolding $\text{zeta_strip_region}'_def$ **using** $h(1) \ h(2)$ **by** *auto*
moreover hence $1 - 1 / 7 * C_1 / \ln(x + 3) \leq 1 - C_4 / \ln x$ **using** $h(4)$ **by** *auto*
ultimately have $\|\text{logderiv zeta}(\text{Complex}(1 - C_4 / \ln x) t)\| \leq C_2 * (\ln(|\text{Im}(\text{Complex}(1 - C_4 / \ln x) t)| + 3))^2$
using *strip_in_logderiv_zeta_region* [**where** $?\sigma = 1 - C_4 / \ln x$ **and** $?T = x$]
by (*intro logderiv_zeta_region_estimate*) *auto*
also have $\dots \leq C_2 * (\ln(x + 3))^2$
by (*intro mult_left_mono, subst power2_le_iff_abs_le*)
(use C2_gt_zero h(2) h(3) in auto)
also have $\dots \leq C_5 * (\ln x)^2$ **by** (*rule h(5)*)
finally show *?thesis* .
qed
hence $\forall_F x \text{ in } \text{at_top}. \forall t. C_3 \leq |t| \implies |t| \leq x$
 $\implies 1 < x \implies C_4 / \ln x \leq C_1 / (7 * \ln(x + 3))$
 $\implies C_2 * (\ln(x + 3))^2 \leq C_5 * (\ln x)^2$
 $\implies \|\text{logderiv zeta}(\text{Complex}(1 - C_4 / \ln x) t)\| \leq C_5 * (\ln x)^2$
by (*intro eventuallyI*) *blast*
moreover have $\forall_F x \text{ in } \text{at_top}. (1 :: \text{real}) < x$ **by** *auto*
ultimately have $1: \forall_F x \text{ in } \text{at_top}. \forall t. C_3 \leq |t| \implies |t| \leq x$
 $\implies \|\text{logderiv zeta}(\text{Complex}(1 - C_4 / \ln x) t)\| \leq C_5 * (\ln x)^2$
using C_4_prop hC_5 **by** *eventually_elim blast*
define f **where** $f x \equiv 1 - C_4 / \ln x$ **for** x
define g **where** $g x t \equiv \text{Complex}(f x) t$ **for** $x \ t$
let $?P = \lambda x t. \|\text{logderiv zeta}(g x t)\| \leq M + \ln x / C_4$
have $\alpha < 1$ **using** $H\alpha$ **by** *auto*

hence $\forall_F x$ in *at_top*. $\alpha \leq f x$ **unfolding** *f_def* **using** *C4_gt_zero* **by** *real_asymp*
 moreover have *f_lt_1*: $\forall_F x$ in *at_top*. $f x < 1$ **unfolding** *f_def* **using** *C4_gt_zero* **by** *real_asymp*
 ultimately have $\forall_F x$ in *at_top*. $\forall t. |t| \leq C_3 \longrightarrow g x t \in K' - \{1\}$
unfolding *g_def* *K'_def* **by** *eventually_elim* (*auto simp add: in_cbox_complex_iff_legacy_Complex_simps*)
 moreover have $\|\logderiv zeta (g x t)\| \leq M + 1 / (1 - f x)$
 when *h*: $g x t \in K' - \{1\}$ $f x < 1$ **for** *x t*

proof –

from *h(1)* **have** *ne_1*: $g x t \neq 1$ **by** *auto*

hence $\|\logderiv zeta (g x t)\| = \|\logderiv zeta' (g x t) - 1 / (g x t - 1)\|$

using *h(1) nonzero*

by (*subst logderiv_zeta_eq_zeta'*)

(*auto simp add: zeta_eq_zero_iff_zeta' [symmetric]*)

also have $\dots \leq \|\logderiv zeta' (g x t)\| + \|1 / (g x t - 1)\|$ **by** (*rule norm_triangle_ineq4*)

also have $\dots \leq M + 1 / (1 - f x)$

proof –

have $\|\logderiv zeta' (g x t)\| \leq M$ **using** *that* **by** (*auto intro: hM*)

moreover have $|Re (g x t - 1)| \leq \|g x t - 1\|$ **by** (*rule abs_Re_le_cmod*)

hence $\|1 / (g x t - 1)\| \leq 1 / (1 - f x)$

using *ne_1 h(2)*

by (*auto simp add: norm_divide g_def*)

intro!: *divide_left_mono mult_pos_pos*)

ultimately show *?thesis* **by** *auto*

qed

finally show *?thesis* .

qed

hence $\forall_F x$ in *at_top*. $\forall t. f x < 1$

$\longrightarrow g x t \in K' - \{1\}$

$\longrightarrow \|\logderiv zeta (g x t)\| \leq M + 1 / (1 - f x)$ **by** *auto*

ultimately have $\forall_F x$ in *at_top*. $\forall t. |t| \leq C_3 \longrightarrow \|\logderiv zeta (g x t)\| \leq M + 1 / (1 - f x)$

using *f_lt_1* **by** *eventually_elim blast*

hence $\forall_F x$ in *at_top*. $\forall t. |t| \leq C_3 \longrightarrow \|\logderiv zeta (g x t)\| \leq M + \ln x / C_4$ **unfolding** *f_def* **by** *auto*

moreover have $\forall_F x$ in *at_top*. $M + \ln x / C_4 \leq C_5 * (\ln x)^2$ **using** *C4_gt_zero* *C5_gt_zero* **by** *real_asymp*

ultimately have *2*: $\forall_F x$ in *at_top*. $\forall t. |t| \leq C_3 \longrightarrow \|\logderiv zeta (g x t)\| \leq C_5 * (\ln x)^2$ **by** *eventually_elim auto*

show *?thesis*

proof (*unfold C5_prop_def, intro exI conjI*)

show $0 < C_5$ **by** (*rule C5_gt_zero*)**+**

have $\forall_F x$ in *at_top*. $\forall t. C_3 \leq |t| \vee |t| \leq C_3$

by (*rule eventuallyI*) *auto*

with *1 2* **show** $\forall_F x$ in *at_top*. $\forall t. |t| \leq x \longrightarrow \|\logderiv zeta (\text{Complex } (1 - C_4 / \ln x) t)\| \leq C_5 * (\ln x)^2$

unfolding *f_def* *g_def* **by** *eventually_elim blast*

qed

qed

definition *C5* **where** $C_5 \equiv \text{SOME } C_5$. *C5_prop* *C5*

lemma

C5_gt_zero: $0 < C_5$ (**is** *?prop_1*) **and**

logderiv_zeta_bound_vertical:

$\forall_F x$ in *at_top*. $\forall t. |t| \leq x$

$\longrightarrow \|\logderiv zeta (\text{Complex } (1 - C_4 / \ln x) t)\| \leq C_5 * (\ln x)^2$ (**is** *?prop_2*)

proof –

```

have C5_prop C5_unfolding C5_def
  by (rule someI_ex) (rule logderiv_zeta_bound_vertical')
thus ?prop_1 ?prop_2 unfolding C5_prop_def by auto
qed

```

7 Deducing prime number theorem using Perron's formula

```

locale prime_number_theorem =

```

```

  fixes c ε :: real
  assumes Hc: 0 < c and Hc': c * c < 2 * C4 and Hε: 0 < ε 2 * ε < c
begin
notation primes_psi (ψ)
definition H where H x ≡ exp (c / 2 * (ln x) powr (1 / 2)) for x :: real
definition T where T x ≡ exp (c * (ln x) powr (1 / 2)) for x :: real
definition a where a x ≡ 1 - C4 / (c * (ln x) powr (1 / 2)) for x :: real
definition b where b x ≡ 1 + 1 / (ln x) for x :: real
definition B where B x ≡ 5 / 4 * ln x for x :: real
definition f where f x s ≡ x powr s / s * logderiv zeta s for x :: real and s :: complex
definition R where R x ≡
  x powr (b x) * H x * B x / T x + 3 * (2 + ln (T x / b x))
  * (∑ n | x - x / H x ≤ n ∧ n ≤ x + x / H x. ||fds_nth (fds mangoldt_complex) n||) for x :: real
definition Rc' where Rc' ≡ O(λx. x * exp (- (c / 2 - ε) * ln x powr (1 / 2)))
definition Rc where Rc ≡ O(λx. x * exp (- (c / 2 - 2 * ε) * ln x powr (1 / 2)))
definition z1 where z1 x ≡ Complex (a x) (- T x) for x
definition z2 where z2 x ≡ Complex (b x) (- T x) for x
definition z3 where z3 x ≡ Complex (b x) (T x) for x
definition z4 where z4 x ≡ Complex (a x) (T x) for x
definition rect where rect x ≡ cbox (z1 x) (z3 x) for x
definition rect' where rect' x ≡ rect x - {1} for x
definition Pt where Pt x t ≡ linepath (Complex (a x) t) (Complex (b x) t) for x t
definition P1 where P1 x ≡ linepath (z1 x) (z4 x) for x
definition P2 where P2 x ≡ linepath (z2 x) (z3 x) for x
definition P3 where P3 x ≡ Pt x (- T x) for x
definition P4 where P4 x ≡ Pt x (T x) for x
definition Pr where Pr x ≡ rectpath (z1 x) (z3 x) for x

```

```

lemma Rc_eq_rem_est:

```

```

  Rc = rem_est (c / 2 - 2 * ε) (1 / 2) 0
proof -
  have *: ∀_F x :: real in at_top. 0 < ln (ln x) by real_asymp
  show ?thesis unfolding Rc_def rem_est_def
    by (rule landau_o.big.cong) (use * in eventually_elim, auto)
qed

```

```

lemma residue_f:

```

```

  residue (f x) 1 = - x
proof -
  define A where A ≡ box (Complex 0 (- 1 / 2)) (Complex 2 (1 / 2))
  have hA: 0 ∉ A 1 ∈ A open A
    unfolding A_def by (auto simp add: mem_box Basis_complex_def)
  have zeta' s ≠ 0 when s ∈ A for s
  proof -
    have s ≠ 1 ⇒ zeta s ≠ 0
      using that unfolding A_def
      by (intro zeta_nonzero_small_imag)

```

(auto simp add: mem_box Basis_complex_def)
 thus ?thesis by (subst zeta'_eq_zero_iff) auto
 qed
 hence h: ($\lambda s. x \text{ powr } s / s * \text{logderiv } zeta' s$) holomorphic_on A
 by (intro holomorphic_intros) (use hA in auto)
 have h': ($\lambda s. x \text{ powr } s / (s * (s - 1))$) holomorphic_on A - {1}
 by (auto intro!: holomorphic_intros) (use hA in auto)
 have s_ne_1: $\forall_F s :: \text{complex in at } 1. s \neq 1$
 by (subst eventually_at_filter) auto
 moreover have $\forall_F s \text{ in at } 1. zeta s \neq 0$
 by (intro non_zero_neighbour_pole_is_pole_zeta)
 ultimately have $\forall_F s \text{ in at } 1. \text{logderiv } zeta s = \text{logderiv } zeta' s - 1 / (s - 1)$
 by eventually_elim (rule logderiv_zeta_eq_zeta')
 moreover have
 $f x s = x \text{ powr } s / s * \text{logderiv } zeta' s - x \text{ powr } s / s / (s - 1)$
 when $\text{logderiv } zeta s = \text{logderiv } zeta' s - 1 / (s - 1)$ $s \neq 0$ $s \neq 1$ for $s :: \text{complex}$
 unfolding f_def by (subst that(1)) (insert that, auto simp add: field_simps)
 hence $\forall_F s :: \text{complex in at } 1. s \neq 0 \longrightarrow s \neq 1$
 $\longrightarrow \text{logderiv } zeta s = \text{logderiv } zeta' s - 1 / (s - 1)$
 $\longrightarrow f x s = x \text{ powr } s / s * \text{logderiv } zeta' s - x \text{ powr } s / s / (s - 1)$
 by (intro eventuallyI) blast
 moreover have $\forall_F s :: \text{complex in at } 1. s \neq 0$
 by (subst eventually_at_topological)
 (intro exI [of UNIV - {0}], auto)
 ultimately have $\forall_F s :: \text{complex in at } 1. f x s = x \text{ powr } s / s * \text{logderiv } zeta' s - x \text{ powr } s / s / (s - 1)$
 using s_ne_1 by eventually_elim blast
 hence $\text{residue } (f x) 1 = \text{residue } (\lambda s. x \text{ powr } s / s * \text{logderiv } zeta' s - x \text{ powr } s / s / (s - 1)) 1$
 by (intro residue_cong refl)
 also have $\dots = \text{residue } (\lambda s. x \text{ powr } s / s * \text{logderiv } zeta' s) 1 - \text{residue } (\lambda s. x \text{ powr } s / s / (s - 1)) 1$
 by (subst residue_diff [where ?s = A]) (use h h' hA in auto)
 also have $\dots = - x$
 proof -
 have $\text{residue } (\lambda s. x \text{ powr } s / s * \text{logderiv } zeta' s) 1 = 0$
 by (rule residue_holo [where ?s = A]) (use hA h in auto)
 moreover have $\text{residue } (\lambda s. x \text{ powr } s / s / (s - 1)) 1 = (x :: \text{complex}) \text{ powr } 1 / 1$
 by (rule residue_simple [where ?s = A]) (use hA in (auto intro!: holomorphic_intros))
 ultimately show ?thesis by auto
 qed
 finally show ?thesis .
 qed

lemma rect_in_strip:

$\text{rect } x - \{1\} \subseteq \text{zeta_strip_region } (a x) (T x)$
 unfolding rect_def zeta_strip_region_def z1_def z3_def
 by (auto simp add: in_cbox_complex_iff)

lemma rect_in_strip':

$\{s \in \text{rect } x. C_3 \leq |\text{Im } s|\} \subseteq \text{zeta_strip_region}' (a x) (T x)$
 unfolding rect_def zeta_strip_region'_def z1_def z3_def
 using C3_gt_zero by (auto simp add: in_cbox_complex_iff)

lemma

$\text{rect}'_{\text{in_zerofree}}: \forall_F x \text{ in at_top}. \text{rect}' x \subseteq \text{zeta_zerofree_region}$ and
 $\text{rect}_{\text{in_logderiv_zeta}}: \forall_F x \text{ in at_top}. \{s \in \text{rect } x. C_3 \leq |\text{Im } s|\} \subseteq \text{logderiv_zeta_region}$

proof (goal_cases)

case 1 have

$\forall_F x \text{ in } at_top. C_4 / \ln x \leq C_1 / (7 * \ln (x + 3))$ **by** (rule C_4_prop)

moreover have $LIM x \text{ at_top. } exp (c * (\ln x) \text{ powr } (1 / 2)) :> at_top$ **using** Hc **by** $real_asympt$
ultimately have h :

$\forall_F x \text{ in } at_top. C_4 / \ln (exp (c * (\ln x) \text{ powr } (1 / 2)))$
 $\leq C_1 / (7 * \ln (exp (c * (\ln x) \text{ powr } (1 / 2)) + 3))$ (**is eventually** $?P _$)
by (rule $eventually_compose_filterlim$)

moreover have

$?P x \implies zeta_strip_region (a x) (T x) \subseteq zeta_zerofree_region$
(**is** $_ \implies ?Q$) **for** x **unfolding** $T_def a_def$

by (intro $strip_in_zerofree_region strip_condition_imp$) **auto**

hence $\forall_F x \text{ in } at_top. ?P x \longrightarrow ?Q x$ **by** (intro $eventuallyI$) **blast**

ultimately show $?case$ **unfolding** $rect'_def$ **by** $eventually_elim$ (use $rect_in_strip$ **in** $auto$)

case 2 from h **have**

$?P x \implies zeta_strip_region' (a x) (T x) \subseteq logderiv_zeta_region$
(**is** $_ \implies ?Q$) **for** x **unfolding** $T_def a_def$

by (intro $strip_in_logderiv_zeta_region$) **auto**

hence $\forall_F x \text{ in } at_top. ?P x \longrightarrow ?Q x$ **by** (intro $eventuallyI$) **blast**

thus $?case$ **using** h **by** $eventually_elim$ (use $rect_in_strip'$ **in** $auto$)

qed

lemma $zeta_nonzero_in_rect$:

$\forall_F x \text{ in } at_top. \forall s. s \in rect' x \longrightarrow zeta s \neq 0$

using $rect'_in_zerofree$ **by** $eventually_elim$ (use $zeta_zerofree_region$ **in** $auto$)

lemma $zero_notin_rect$: $\forall_F x \text{ in } at_top. 0 \notin rect' x$

proof –

have $\forall_F x \text{ in } at_top. C_4 / (c * (\ln x) \text{ powr } (1 / 2)) < 1$

using Hc **by** $real_asympt$

thus $?thesis$

unfolding $rect'_def rect_def z1_def z4_def T_def a_def$

by $eventually_elim$ ($simp$ add : $in_cbox_complex_iff$)

qed

lemma $f_analytic$:

$\forall_F x \text{ in } at_top. f x \text{ analytic_on } rect' x$

using $zeta_nonzero_in_rect zero_notin_rect$ **unfolding** f_def

by $eventually_elim$ (intro $analytic_intros$, $auto$ $simp$: $rect'_def$)

lemma $path_image_in_rect_1$:

assumes $0 \leq T x \wedge a x \leq b x$

shows $path_image (P_1 x) \subseteq rect x \wedge path_image (P_2 x) \subseteq rect x$

unfolding $P1_def P2_def rect_def z1_def z2_def z3_def z4_def$

by ($simp$, intro $conjI$ $closed_segment_subset$)

($insert$ $assms$, $auto$ $simp$ add : $in_cbox_complex_iff$)

lemma $path_image_in_rect_2$:

assumes $0 \leq T x \wedge a x \leq b x \wedge t \in \{-T x..T x\}$

shows $path_image (P_t x t) \subseteq rect x$

unfolding $P_t_def rect_def z1_def z3_def$

by ($simp$, intro $conjI$ $closed_segment_subset$)

($insert$ $assms$, $auto$ $simp$ add : $in_cbox_complex_iff$)

definition $path_in_rect'$ **where**

$path_in_rect' x \equiv$

$path_image (P_1 x) \subseteq rect' x \wedge path_image (P_2 x) \subseteq rect' x \wedge$
 $path_image (P_3 x) \subseteq rect' x \wedge path_image (P_4 x) \subseteq rect' x$

lemma *path_image_in_rect'*:

assumes $0 < T x \wedge a x < 1 \wedge 1 < b x$

shows *path_in_rect' x*

proof –

have $path_image (P_1 x) \subseteq rect x \wedge path_image (P_2 x) \subseteq rect x$

by (*rule path_image_in_rect_1*) (*use assms in auto*)

moreover have $path_image (P_3 x) \subseteq rect x \wedge path_image (P_4 x) \subseteq rect x$

unfolding *P3_def P4_def*

by (*intro path_image_in_rect_2, (use assms in auto)[1]*)+

moreover have

$1 \notin path_image (P_1 x) \wedge 1 \notin path_image (P_2 x) \wedge$

$1 \notin path_image (P_3 x) \wedge 1 \notin path_image (P_4 x)$

unfolding *P1_def P2_def P3_def P4_def P_t_def z1_def z2_def z3_def z4_def* **using** *assms*

by (*auto simp add: closed_segment_def legacy_Complex_simps field_simps*)

ultimately show *?thesis unfolding path_in_rect'_def rect'_def* **by** *blast*

qed

lemma *asympt_1*:

$\forall_F x$ *in at_top*. $0 < T x \wedge a x < 1 \wedge 1 < b x$

unfolding *T_def a_def b_def*

by (*intro eventually_conj, insert Hc C4_gt_zero*) (*real_asymp*)+

lemma *f_continuous_on*:

$\forall_F x$ *in at_top*. $\forall A \subseteq rect' x$. *continuous_on A (f x)*

using *f_analytic*

by (*eventually_elim, safe*)

(*intro holomorphic_on_imp_continuous_on analytic_imp_holomorphic,*
elim analytic_on_subset)

lemma *contour_integrability*:

$\forall_F x$ *in at_top*.

$f x$ *contour_integrable_on* *P1 x* \wedge $f x$ *contour_integrable_on* *P2 x* \wedge

$f x$ *contour_integrable_on* *P3 x* \wedge $f x$ *contour_integrable_on* *P4 x*

proof –

have $\forall_F x$ *in at_top*. *path_in_rect' x*

using *asympt_1* **by** *eventually_elim (rule path_image_in_rect')*

thus *?thesis* **using** *f_continuous_on*

unfolding *P1_def P2_def P3_def P4_def P_t_def path_in_rect'_def*

by *eventually_elim*

(*intro conjI contour_integrable_continuous_linepath,*

fold z1_def z2_def z3_def z4_def, auto)

qed

lemma *contour_integral_rectpath'*:

assumes $f x$ *analytic_on* (*rect' x*) $0 < T x \wedge a x < 1 \wedge 1 < b x$

shows *contour_integral (P_r x) (f x) = - 2 * pi * i * x*

proof –

define *z* **where** $z \equiv (1 + b x) / 2$

have *H**z*: $z \in box (z_1 x) (z_3 x)$

unfolding *z1_def z3_def z_def* **using** *assms(2)*

by (*auto simp add: mem_box Basis_complex_def*)

have *H**z'*: $z \neq 1$ **unfolding** *z_def* **using** *assms(2)* **by** *auto*

have *connected* (*rect' x*)
proof –
have *box_nonempty*: $\text{box } (z_1 x) (z_3 x) \neq \{\}$ **using** *Hz* **by** *auto*
hence *aff_dim* (*closure* ($\text{box } (z_1 x) (z_3 x)$)) = 2
by (*subst closure_aff_dim*, *subst aff_dim_open*) *auto*
thus *?thesis*
unfolding *rect'_def* **using** *box_nonempty*
by (*subst (asm) closure_box*)
(auto intro: connected_punctured_convex simp add: rect_def)
qed
moreover **have** *Hz''*: $z \in \text{rect}' x$
unfolding *rect'_def* *rect_def* **using** *box_subset_cbox Hz Hz'* **by** *auto*
ultimately obtain *T* **where** *hT*:
f x holomorphic_on T open T rect' x \subseteq T connected T
using *analytic_on_holomorphic_connected assms(1)* **by** (*metis dual_order.refl*)
define *U* **where** $U \equiv T \cup \text{box } (z_1 x) (z_3 x)$
have *one_in_box*: $1 \in \text{box } (z_1 x) (z_3 x)$
unfolding *z1_def z3_def z_def* **using** *assms(2)* **by** (*auto simp add: mem_box Basis_complex_def*)
have *contour_integral* ($P_r x$) (*f x*) = $2 * \pi * i * (\sum_{s \in \{1\}} \text{winding_number } (P_r x) s * \text{residue } (f x) s)$
proof (*rule Residue_theorem*)
show *finite* $\{1\}$ *valid_path* ($P_r x$) *pathfinish* ($P_r x$) = *pathstart* ($P_r x$)
unfolding *P_r_def* **by** *auto*
show *open U* **unfolding** *U_def* **using** *hT(2)* **by** *auto*
show *connected U* **unfolding** *U_def*
by (*intro connected_Un hT(4) convex_connected*)
(use Hz Hz'' hT(3) in auto)
have *f x holomorphic_on* $\text{box } (z_1 x) (z_3 x) - \{1\}$
by (*rule holomorphic_on_subset, rule analytic_imp_holomorphic, rule assms(1)*)
(unfold rect'_def rect_def, use box_subset_cbox in auto)
hence *f x holomorphic_on* $((T - \{1\}) \cup (\text{box } (z_1 x) (z_3 x) - \{1\}))$
by (*intro holomorphic_on_Un*) *(use hT(1) hT(2) in auto)*
moreover **have** $\dots = U - \{1\}$ **unfolding** *U_def* **by** *auto*
ultimately show *f x holomorphic_on* $U - \{1\}$ **by** *auto*
have *Hz*: $\text{Re } (z_1 x) \leq \text{Re } (z_3 x) \text{ Im } (z_1 x) \leq \text{Im } (z_3 x)$
unfolding *z1_def z3_def* **using** *assms(2)* **by** *auto*
have *path_image* ($P_r x$) = $\text{rect } x - \text{box } (z_1 x) (z_3 x)$
unfolding *rect_def P_r_def*
by (*intro path_image_rectpath_cbox_minus_box Hz*)
thus *path_image* ($P_r x$) $\subseteq U - \{1\}$
using *one_in_box hT(3) U_def* **unfolding** *rect'_def* **by** *auto*
have *hU'*: $\text{rect } x \subseteq U$
using *hT(3) one_in_box* **unfolding** *U_def rect'_def* **by** *auto*
show $\forall z. z \notin U \longrightarrow \text{winding_number } (P_r x) z = 0$
using *Hz P_r_def hU' rect_def winding_number_rectpath_outside* **by** *fastforce*
qed
also **have** $\dots = -2 * \pi * i * x$ **unfolding** *P_r_def*
by (*simp add: residue_f, subst winding_number_rectpath, auto intro: one_in_box*)
finally show *?thesis* .
qed

lemma *contour_integral_rectpath*:

$\forall_F x \text{ in } \text{at_top}. \text{contour_integral } (P_r x) (f x) = -2 * \pi * i * x$
using *f_analytic asymp_1* **by** *eventually_elim (rule contour_integral_rectpath')*

lemma *valid_paths*:

valid_path ($P_1 x$) *valid_path* ($P_2 x$) *valid_path* ($P_3 x$) *valid_path* ($P_4 x$)
unfolding P_1_def P_2_def P_3_def P_4_def P_t_def **by** *auto*

lemma *integral_rectpath_split*:

assumes $f x$ *contour_integrable_on* $P_1 x \wedge f x$ *contour_integrable_on* $P_2 x \wedge$
 $f x$ *contour_integrable_on* $P_3 x \wedge f x$ *contour_integrable_on* $P_4 x$
shows *contour_integral* ($P_3 x$) ($f x$) + *contour_integral* ($P_2 x$) ($f x$)
– *contour_integral* ($P_4 x$) ($f x$) – *contour_integral* ($P_1 x$) ($f x$) = *contour_integral* ($P_r x$) ($f x$)

proof –

define Q_1 **where** $Q_1 \equiv \text{linepath } (z_3 x) (z_4 x)$
define Q_2 **where** $Q_2 \equiv \text{linepath } (z_4 x) (z_1 x)$
have Q_eq : $Q_1 = \text{reversepath } (P_4 x)$ $Q_2 = \text{reversepath } (P_1 x)$
unfolding Q_1_def Q_2_def P_1_def P_4_def P_t_def **by** (*fold* z_3_def z_4_def) *auto*
hence *contour_integral* Q_1 ($f x$) = – *contour_integral* ($P_4 x$) ($f x$)
contour_integral Q_2 ($f x$) = – *contour_integral* ($P_1 x$) ($f x$)
by (*auto intro: contour_integral_reversepath valid_paths*)
moreover have *contour_integral* ($P_3 x$ +++ $P_2 x$ +++ Q_1 +++ Q_2) ($f x$)
= *contour_integral* ($P_3 x$) ($f x$) + *contour_integral* ($P_2 x$) ($f x$)
+ *contour_integral* Q_1 ($f x$) + *contour_integral* Q_2 ($f x$)

proof –

have 1: *pathfinish* ($P_2 x$) = *pathstart* (Q_1 +++ Q_2) *pathfinish* Q_1 = *pathstart* Q_2
unfolding P_2_def Q_1_def Q_2_def **by** *auto*
have 2: *valid_path* Q_1 *valid_path* Q_2 **unfolding** Q_1_def Q_2_def **by** *auto*
have 3: $f x$ *contour_integrable_on* $P_1 x$ $f x$ *contour_integrable_on* $P_2 x$
 $f x$ *contour_integrable_on* $P_3 x$ $f x$ *contour_integrable_on* $P_4 x$
 $f x$ *contour_integrable_on* Q_1 $f x$ *contour_integrable_on* Q_2
using *assms* **by** (*auto simp add: Q_eq intro: contour_integrable_reversepath valid_paths*)
show *?thesis* **by** (*subst contour_integral_join |*
auto intro: valid_paths valid_path_join contour_integrable_joinI 1 2 3)+

qed

ultimately show *?thesis*

unfolding P_r_def z_1_def z_3_def *rectpath_def*
by (*simp add: Let_def, fold P_t_def P_3_def z_1_def z_2_def z_3_def z_4_def*)
(*fold P_2_def Q_1_def Q_2_def, auto*)

qed

lemma P_2_eq :

$\forall_F x$ *in* *at_top*. *contour_integral* ($P_2 x$) ($f x$) + $2 * \pi * i * x$
= *contour_integral* ($P_1 x$) ($f x$) – *contour_integral* ($P_3 x$) ($f x$) + *contour_integral* ($P_4 x$) ($f x$)

proof –

have $\forall_F x$ *in* *at_top*. *contour_integral* ($P_3 x$) ($f x$) + *contour_integral* ($P_2 x$) ($f x$)
– *contour_integral* ($P_4 x$) ($f x$) – *contour_integral* ($P_1 x$) ($f x$) = – $2 * \pi * i * x$
using *contour_integrability contour_integral_rectpath asymp_1 f_analytic*
by *eventually_elim (metis integral_rectpath_split)*
thus *?thesis* **by** (*auto simp add: field_simps*)

qed

lemma *estimation_P1*:

($\lambda x. \| \text{contour_integral } (P_1 x) (f x) \|$) $\in R_c$

proof –

define r **where** $r x \equiv$
 $C_5 * (c * (\ln x) \text{ powr } (1 / 2))^2 * x \text{ powr } a * \ln (1 + T x / a x)$ **for** x
note *logderiv_zeta_bound_vertical*
moreover have *LIM* x *at_top*. $T x$ \rightarrow *at_top*

unfolding T_def **using** Hc **by** $real_asympt$
ultimately have $\forall_F x$ *in* at_top . $\forall t. |t| \leq T x$
 $\longrightarrow \|logderiv\ zeta\ (Complex\ (1 - C_4 / \ln\ (T\ x))\ t)\| \leq C_5 * (\ln\ (T\ x))^2$
unfolding a_def **by** $(rule\ eventually_compose_filterlim)$
hence $\forall_F x$ *in* at_top . $\forall t. |t| \leq T x$
 $\longrightarrow \|logderiv\ zeta\ (Complex\ (a\ x)\ t)\| \leq C_5 * (c * (\ln\ x)\ powr\ (1 / 2))^2$
unfolding $a_def\ T_def$ **by** $auto$
moreover have $\forall_F x$ *in* at_top . $(f\ x)$ *contour_integrable_on* $(P_1\ x)$
using $contour_integrability$ **by** $eventually_elim\ auto$
hence $\forall_F x$ *in* at_top . $(\lambda s. logderiv\ zeta\ s * x\ powr\ s / s)$ *contour_integrable_on* $(P_1\ x)$
unfolding f_def **by** $eventually_elim\ (auto\ simp\ add:\ field_simps)$
moreover have $\forall_F x :: real$ *in* at_top . $0 < x$ **by** $auto$
moreover have $\forall_F x$ *in* at_top . $0 < a\ x$ **unfolding** a_def **using** Hc **by** $real_asympt$
ultimately have $\forall_F x$ *in* at_top .
 $\|1 / (2 * pi * i) * contour_integral\ (P_1\ x)\ (\lambda s. logderiv\ zeta\ s * x\ powr\ s / s)\| \leq r\ x$
unfolding $r_def\ P_1_def\ z_1_def\ z_4_def$ **using** $asympt_1$
by $eventually_elim\ (rule\ perron_aux_3',\ auto)$
hence $\forall_F x$ *in* at_top . $\|1 / (2 * pi * i) * contour_integral\ (P_1\ x)\ (f\ x)\| \leq r\ x$
unfolding f_def **by** $eventually_elim\ (auto\ simp\ add:\ mult_ac)$
hence $(\lambda x. \|1 / (2 * pi * i) * contour_integral\ (P_1\ x)\ (f\ x)\|) \in O(r)$
unfolding f_def **by** $(rule\ eventually_le_imp_bigo')$
moreover have $r \in Rc$
proof –
define r_1 **where** $r_1\ x \equiv C_5 * c^2 * \ln\ x * \ln\ (1 + T\ x / a\ x)$ **for** x
define r_2 **where** $r_2\ x \equiv exp\ (a\ x * \ln\ x)$ **for** x
have $r_1 \in O(\lambda x. (\ln\ x)^2)$
unfolding $r_1_def\ T_def\ a_def$ **using** $Hc\ C_5_gt_zero$ **by** $real_asympt$
moreover have $r_2 \in Rc'$
proof –
have $1: \|r_2\ x\| \leq x * exp\ (- (c / 2 - \epsilon) * (\ln\ x)\ powr\ (1 / 2))$
when $h: 0 < x < \ln\ x$ **for** x
proof –
have $a\ x * \ln\ x = \ln\ x + - C_4 / c * (\ln\ x)\ powr\ (1 / 2)$
unfolding a_def **using** $h(2)\ Hc$
by $(auto\ simp\ add:\ field_simps\ powr_add\ [symmetric]\ frac_eq_eq)$
hence $r_2\ x = exp\ (...)$ **unfolding** r_2_def **by** $blast$
also have $... = x * exp\ (- C_4 / c * (\ln\ x)\ powr\ (1 / 2))$
by $(subst\ exp_add)\ (use\ h(1)\ in\ auto)$
also have $... \leq x * exp\ (- (c / 2 - \epsilon) * (\ln\ x)\ powr\ (1 / 2))$
by $(intro\ mult_left_mono,\ subst\ exp_le_cancel_iff,\ intro\ mult_right_mono)$
 $(use\ Hc\ Hc'\ H\epsilon\ C_4_gt_zero\ h\ in\ \langle auto\ simp:\ field_simps\ intro:\ add_increasing2 \rangle)$
finally show $?thesis$ **unfolding** r_2_def **by** $auto$
qed
have $\forall_F x$ *in* at_top . $\|r_2\ x\| \leq x * exp\ (- (c / 2 - \epsilon) * (\ln\ x)\ powr\ (1 / 2))$
using $\ln_asympt_pos\ x_asympt_pos$ **by** $eventually_elim\ (rule\ 1)$
thus $?thesis$ **unfolding** Rc'_def **by** $(rule\ eventually_le_imp_bigo)$
qed
ultimately have $(\lambda x. r_1\ x * r_2\ x)$
 $\in O(\lambda x. (\ln\ x)^2 * (x * exp\ (- (c / 2 - \epsilon) * (\ln\ x)\ powr\ (1 / 2))))$
unfolding Rc'_def **by** $(rule\ landau_o.big.mult)$
moreover have $(\lambda x. (\ln\ x)^2 * (x * exp\ (- (c / 2 - \epsilon) * (\ln\ x)\ powr\ (1 / 2)))) \in Rc$
unfolding Rc_def **using** $Hc\ H\epsilon$
by $(real_asympt\ simp\ add:\ field_simps)$
ultimately have $(\lambda x. r_1\ x * r_2\ x) \in Rc$
unfolding Rc_def **by** $(rule\ landau_o.big.trans)$

moreover have $\forall_F x \text{ in } at_top. r\ x = r_1\ x * r_2\ x$
using $ln_ln_asympt_pos\ ln_asympt_pos\ x_asympt_pos$
unfolding $r_def\ r_1_def\ r_2_def\ a_def\ powr_def\ power2_eq_square$
by $(eventually_elim)\ (simp\ add:\ field_simps\ exp_add\ [symmetric])$
ultimately show *?thesis* **unfolding** Rc_def
using $landau_o.big.ev_eq_trans2$ **by** *auto*
qed
ultimately have $(\lambda x. \|1 / (2 * pi * i) * contour_integral\ (P_1\ x)\ (f\ x)\|) \in Rc$
unfolding Rc_def **by** $(rule\ landau_o.big_trans)$
thus *?thesis* **unfolding** Rc_def **by** $(simp\ add:\ norm_divide)$
qed

lemma $estimation_P_t'$:
assumes h :
 $1 < x \wedge \max\ 1\ C_3 \leq T\ x\ a\ x < 1 \wedge 1 < b\ x$
 $\{s \in rect\ x. C_3 \leq |Im\ s|\} \subseteq logderiv_zeta_region$
 $f\ x\ contour_integrable_on\ P_3\ x \wedge f\ x\ contour_integrable_on\ P_4\ x$
and $Ht: |t| = T\ x$
shows $\|contour_integral\ (P_t\ x\ t)\ (f\ x)\| \leq C_2 * exp\ 1 * x / T\ x * (ln\ (T\ x + 3))^2 * (b\ x - a\ x)$
proof –
consider $t = T\ x \mid t = -\ T\ x$ **using** Ht **by** *fastforce*
hence $f\ x\ contour_integrable_on\ P_t\ x\ t$
using $Ht\ h(4)$ **unfolding** $P_t_def\ P_3_def\ P_4_def$ **by** *cases\ auto*
moreover have $\|f\ x\ s\| \leq exp\ 1 * x / T\ x * (C_2 * (ln\ (T\ x + 3))^2)$
when $s \in closed_segment\ (Complex\ (a\ x)\ t)\ (Complex\ (b\ x)\ t)$ **for** s
proof –
have $Hs: s \in path_image\ (P_t\ x\ t)$ **using** *that* **unfolding** P_t_def **by** *auto*
have $path_image\ (P_t\ x\ t) \subseteq rect\ x$
by $(rule\ path_image_in_rect_2)$ $(use\ h(2)\ Ht\ in\ auto)$
moreover have $Hs': Re\ s \leq b\ x\ Im\ s = t$
proof –
have $u \leq 1 \implies (1 - u) * a\ x \leq (1 - u) * b\ x$ **for** u
using $h(2)$ **by** $(intro\ mult_left_mono)\ auto$
thus $Re\ s \leq b\ x\ Im\ s = t$
using *that* $h(2)$ **unfolding** $closed_segment_def$
by $(auto\ simp\ add:\ legacy_Complex_simps\ field_simps)$
qed
hence $C_3 \leq |Im\ s|$ **using** $h(1)\ Ht$ **by** *auto*
ultimately have $s \in logderiv_zeta_region$ **using** $Hs\ h(3)$ **by** *auto*
hence $\|logderiv\ zeta\ s\| \leq C_2 * (ln\ (|Im\ s| + 3))^2$
by $(rule\ logderiv_zeta_region_estimate)$
also have $\dots = C_2 * (ln\ (T\ x + 3))^2$ **using** $Hs'(2)\ Ht$ **by** *auto*
also have $\|x\ powr\ s / s\| \leq exp\ 1 * x / T\ x$
proof –
have $\|x\ powr\ s\| = Re\ x\ powr\ Re\ s$ **using** $h(1)$ **by** $(intro\ norm_powr_real_powr)\ auto$
also have $\dots = x\ powr\ Re\ s$ **by** *auto*
also have $\dots \leq x\ powr\ b\ x$ **by** $(intro\ powr_mono\ Hs')$ $(use\ h(1)\ in\ auto)$
also have $\dots = exp\ 1 * x$
using $h(1)$ **unfolding** $powr_def\ b_def$ **by** $(auto\ simp\ add:\ field_simps\ exp_add)$
finally have $\|x\ powr\ s\| \leq exp\ 1 * x$
moreover have $T\ x \leq \|s\|$ **using** $abs\ Im_le_cmod\ [of\ s]\ Hs'(2)\ h(1)\ Ht$ **by** *auto*
hence $1: \|x\ powr\ s\| / \|s\| \leq \|x\ powr\ s\| / T\ x$
using $h(1)$ **by** $(intro\ divide_left_mono\ mult_pos_pos)\ auto$
ultimately have $\dots \leq exp\ 1 * x / T\ x$
by $(intro\ divide_right_mono)\ (use\ h(1)\ in\ auto)$

thus *?thesis using 1 by (subst norm_divide) linarith*
 qed
 ultimately show *?thesis unfolding f_def*
 by *(subst norm_mult, intro mult_mono, auto)*
(metis norm_ge_zero order.trans)
 qed
 ultimately have $\| \text{contour_integral } (P_t \ x \ t) \ (f \ x) \|$
 $\leq \exp 1 * x / T \ x * (C_2 * (\ln (T \ x + 3)))^2 * \| \text{Complex } (b \ x) \ t - \text{Complex } (a \ x) \ t \|$
 unfolding *P_t_def*
 by *(intro contour_integral_bound_linepath)*
(use C2_gt_zero h(1) in auto)
 also have $\dots = C_2 * \exp 1 * x / T \ x * (\ln (T \ x + 3))^2 * (b \ x - a \ x)$
 using *h(2) by (simp add: legacy_Complex_simps)*
 finally show *?thesis .*
 qed

lemma estimation_Pt:
 $(\lambda x. \| \text{contour_integral } (P_3 \ x) \ (f \ x) \|) \in Rc \wedge$
 $(\lambda x. \| \text{contour_integral } (P_4 \ x) \ (f \ x) \|) \in Rc$
proof –
 define *r where r x ≡ C₂ * exp 1 * x / T x * (ln (T x + 3))² * (b x - a x) for x*
 define *p where p x ≡ \| contour_integral (P₃ x) (f x) \| ≤ r x ∧ \| contour_integral (P₄ x) (f x) \| ≤ r x*
for x
 have $\forall_F \ x \ \text{in } \text{at_top}. \ 1 < x \wedge \max 1 \ C_3 \leq T \ x$
 unfolding *T_def* by *(rule eventually_conj) (simp, use Hc in real_asymp)*
 hence $\forall_F \ x \ \text{in } \text{at_top}. \ \forall t. \ |t| = T \ x \longrightarrow \| \text{contour_integral } (P_t \ x \ t) \ (f \ x) \| \leq r \ x$ **(is eventually ?P _)**
 unfolding *r_def* using *asymp_1 rect_in_logderiv_zeta contour_integrability*
 by *eventually_elim (use estimation_Pt' in blast)*
 moreover have $\bigwedge x. \ ?P \ x \implies 0 < T \ x \implies p \ x$
 unfolding *p_def P₃_def P₄_def* by *auto*
 hence $\forall_F \ x \ \text{in } \text{at_top}. \ ?P \ x \longrightarrow 0 < T \ x \longrightarrow p \ x$
 by *(intro eventuallyI) blast*
 ultimately have $\forall_F \ x \ \text{in } \text{at_top}. \ p \ x$ using *asymp_1* by *eventually_elim blast*
 hence $\forall_F \ x \ \text{in } \text{at_top}. \ \| \| \text{contour_integral } (P_3 \ x) \ (f \ x) \| \| \leq 1 * \| r \ x \| \wedge$
 $\| \| \text{contour_integral } (P_4 \ x) \ (f \ x) \| \| \leq 1 * \| r \ x \|$
 unfolding *p_def* by *eventually_elim auto*
 hence $(\lambda x. \| \text{contour_integral } (P_3 \ x) \ (f \ x) \|) \in O(r) \wedge (\lambda x. \| \text{contour_integral } (P_4 \ x) \ (f \ x) \|) \in O(r)$
 by *(subst (asm) eventually_conj_iff, blast)+*
 moreover have $r \in Rc$
 unfolding *r_def Rc_def a_def b_def T_def* using *Hc Hε*
 by *(real_asymp simp add: field_simps)*
 ultimately show *?thesis*
 unfolding *Rc_def* using *landau_o.big_trans* by *blast*
 qed

lemma Re_path_P2:
 $\bigwedge z. \ z \in \text{path_image } (P_2 \ x) \implies \text{Re } z = b \ x$
 unfolding *P₂_def z₂_def z₃_def*
 by *(auto simp add: closed_segment_def legacy_Complex_simps field_simps)*

lemma estimation_P2:
 $(\lambda x. \| 1 / (2 * pi * i) * \text{contour_integral } (P_2 \ x) \ (f \ x) + x \|) \in Rc$
proof –
 define *r where r x ≡ \| contour_integral (P₁ x) (f x) \| +*

$\| \text{contour_integral } (P_3 x) (f x) \| + \| \text{contour_integral } (P_4 x) (f x) \|$ **for** x
have $[simp]: \|a - b + c\| \leq \|a\| + \|b\| + \|c\|$ **for** $a b c :: \text{complex}$
using $\text{adhoc_norm_triangle norm_triangle_ineq4}$ **by** blast
have $\forall_F x \text{ in at_top. } \| \text{contour_integral } (P_2 x) (f x) + 2 * \text{pi} * i * x \| \leq r x$
unfolding r_def **using** P_2_eq **by** $\text{eventually_elim auto}$
hence $(\lambda x. \| \text{contour_integral } (P_2 x) (f x) + 2 * \text{pi} * i * x \|) \in O(r)$
by $(\text{rule eventually_le_imp_bigo})$
moreover **have** $r \in Rc$
using $\text{estimation_}P_1 \text{ estimation_}P_t$
unfolding $r_def Rc_def$ **by** $(\text{intro sum_in_bigo}) \text{ auto}$
ultimately **have** $(\lambda x. \| \text{contour_integral } (P_2 x) (f x) + 2 * \text{pi} * i * x \|) \in Rc$
unfolding Rc_def **by** $(\text{rule landau_o.big_trans})$
hence $(\lambda x. \|1 / (2 * \text{pi} * i) * (\text{contour_integral } (P_2 x) (f x) + 2 * \text{pi} * i * x)\|) \in Rc$
unfolding Rc_def **by** $(\text{auto simp add: norm_mult norm_divide})$
thus $?thesis$ **by** $(\text{auto simp add: algebra_simps})$

qed

lemma $\text{estimation_}R$:

$R \in Rc$

proof –

define Γ **where** $\Gamma x \equiv \{n :: \text{nat. } x - x / H x \leq n \wedge n \leq x + x / H x\}$ **for** x

have $1: (\lambda x. x \text{ powr } b x * H x * B x / T x) \in Rc$

unfolding $b_def H_def B_def T_def Rc_def$ **using** $Hc H\epsilon$

by $(\text{real_asympt simp add: field_simps})$

have $\| \sum_{n \in \Gamma x} n \| \leq (2 * x / H x + 1) * \ln (x + x / H x)$

when $h: 0 < x - x / H x < x / H x < x / H x \leq \ln (x + x / H x)$ **for** x

proof –

have $\| \sum_{n \in \Gamma x} n \| = (\sum_{n \in \Gamma x} \| \text{fds_nth } (\text{fds mangoldt_complex}) n \|)$

by $\text{simp } (\text{subst abs_of_nonneg, auto intro: sum_nonneg})$

also **have** $\dots = \text{sum mangoldt_real } (\Gamma x)$

by $(\text{subst norm_fds_mangoldt_complex}) (\text{rule refl})$

also **have** $\dots \leq \text{card } (\Gamma x) * \ln (x + x / H x)$

proof $(\text{rule sum_bounded_above})$

fix n **assume** $n \in \Gamma x$

hence $Hn: 0 < n \leq x + x / H x$ **unfolding** Γ_def **using** h **by** auto

hence $\text{mangoldt_real } n \leq \ln n$ **by** $(\text{intro mangoldt_le})$

also **have** $\dots \leq \ln (x + x / H x)$ **using** Hn **by** auto

finally **show** $\text{mangoldt_real } n \leq \ln (x + x / H x)$.

qed

also **have** $\dots \leq (2 * x / H x + 1) * \ln (x + x / H x)$

proof –

have $\Gamma_eq: \Gamma x = \{ \text{nat } \lceil x - x / H x \rceil .. \text{nat } (\lfloor x + x / H x \rfloor + 1) \}$

unfolding Γ_def **by** $(\text{subst nat_le_real_iff}) (\text{subst nat_ceiling_le_eq } [\text{symmetric}], \text{ auto})$

moreover **have** $\text{nat } (\lfloor x + x / H x \rfloor + 1) = \lfloor x + x / H x \rfloor + 1$ **using** $h(1) h(2)$ **by** auto

moreover **have** $\text{nat } \lceil x - x / H x \rceil = \lceil x - x / H x \rceil$ **using** $h(1)$ **by** auto

moreover **have** $\lfloor x + x / H x \rfloor \leq x + x / H x$ **by** (rule floor_le)

moreover **have** $\lceil x - x / H x \rceil \geq x - x / H x$ **by** (rule ceil_ge)

ultimately **have** $(\text{nat } (\lfloor x + x / H x \rfloor + 1) :: \text{real}) - \text{nat } \lceil x - x / H x \rceil \leq 2 * x / H x + 1$ **by**

linarith

hence $\text{card } (\Gamma x) \leq 2 * x / H x + 1$ **using** $h(2)$ **by** $(\text{subst } \Gamma_eq) (\text{auto simp add: of_nat_diff_real})$

thus $?thesis$ **using** $h(3)$ **by** $(\text{rule mult_right_mono})$

qed

finally **show** $?thesis$.

qed

hence $\forall_F x$ in *at_top*.

$$0 < x - x / H x \longrightarrow 0 < x / H x \longrightarrow 0 \leq \ln (x + x / H x)$$

$$\longrightarrow \|\sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|\| \leq (2 * x / H x + 1) * \ln (x + x / H x)$$

by (*intro eventuallyI*) *blast*

moreover have $\forall_F x$ in *at_top*. $0 < x - x / H x$ **unfolding** *H_def* **using** *Hc Hε* **by** *real_asymp*

moreover have $\forall_F x$ in *at_top*. $0 < x / H x$ **unfolding** *H_def* **using** *Hc Hε* **by** *real_asymp*

moreover have $\forall_F x$ in *at_top*. $0 \leq \ln (x + x / H x)$ **unfolding** *H_def* **using** *Hc Hε* **by** *real_asymp*

ultimately have $\forall_F x$ in *at_top*. $\|\sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|\| \leq (2 * x / H x + 1) * \ln (x + x / H x)$

by *eventually_elim blast*

hence $(\lambda x. \sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|) \in O(\lambda x. (2 * x / H x + 1) * \ln (x + x / H x))$

by (*rule eventually_le_imp_bigo*)

moreover have $(\lambda x. (2 * x / H x + 1) * \ln (x + x / H x)) \in Rc'$

unfolding *Rc'_def H_def* **using** *Hc Hε*

by (*real_asymp simp add: field_simps*)

ultimately have $(\lambda x. \sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|) \in Rc'$

unfolding *Rc'_def* **by** (*rule landau_o.big_trans*)

hence $(\lambda x. 3 * (2 + \ln (T x / b x)) * (\sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|))$

$$\in O(\lambda x. 3 * (2 + \ln (T x / b x)) * (x * \exp (- (c / 2 - \varepsilon) * (\ln x) \text{ powr } (1 / 2))))$$

unfolding *Rc'_def* **by** (*intro landau_o.big.mult_left*) *auto*

moreover have $(\lambda x. 3 * (2 + \ln (T x / b x)) * (x * \exp (- (c / 2 - \varepsilon) * (\ln x) \text{ powr } (1 / 2)))) \in Rc$

unfolding *Rc_def T_def b_def* **using** *Hc Hε* **by** (*real_asymp simp add: field_simps*)

ultimately have $2: (\lambda x. 3 * (2 + \ln (T x / b x)) * (\sum_{n \in \Gamma} x. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\|)) \in Rc$

unfolding *Rc_def* **by** (*rule landau_o.big_trans*)

from *1 2* **show** *?thesis* **unfolding** *Rc_def R_def Γ_def* **by** (*rule sum_in_bigo*)

qed

lemma *perron_psi*:

$$\forall_F x$$
 in *at_top*. $\|\psi x + 1 / (2 * \pi * i) * \text{contour_integral} (P_2 x) (f x)\| \leq R x$

proof –

have *Hb*: $\forall_F x$ in *at_top*. $1 < b x$ **unfolding** *b_def* **by** *real_asymp*

hence $\forall_F x$ in *at_top*. $0 < b x$ **by** *eventually_elim auto*

moreover have $\forall_F x$ in *at_top*. $b x \leq T x$ **unfolding** *b_def T_def* **using** *Hc* **by** *real_asymp*

moreover have $\forall_F x$ in *at_top*. $\text{abs_conv_abscissa} (f_{ds_mangoldt_complex}) < \text{ereal} (b x)$

proof –

have $\text{abs_conv_abscissa} (f_{ds_mangoldt_complex}) \leq 1$ **by** (*rule abs_conv_abscissa_mangoldt*)

hence $\forall_F x$ in *at_top*. $1 < b x \longrightarrow \text{abs_conv_abscissa} (f_{ds_mangoldt_complex}) < \text{ereal} (b x)$

by (*auto intro: eventuallyI*)

simp add: le_ereal_less_one_ereal_def)

thus *?thesis* **using** *Hb* **by** (*rule eventually_mp*)

qed

moreover have $\forall_F x$ in *at_top*. $2 \leq H x$ **unfolding** *H_def* **using** *Hc* **by** *real_asymp*

moreover have $\forall_F x$ in *at_top*. $b x + 1 \leq H x$ **unfolding** *b_def H_def* **using** *Hc* **by** *real_asymp*

moreover have $\forall_F x :: \text{real}$ in *at_top*. $2 \leq x$ **by** *auto*

moreover have $\forall_F x$ in *at_top*.

$$(\sum 'n \geq 1. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\| / n \text{ nat_powr } b x) \leq B x$$

(*is eventually ?P ?F*)

proof –

have *?P x* **when** *Hb*: $1 < b x \wedge b x \leq 23 / 20$ **for** *x*

proof –

$$\text{have } (\sum 'n \geq 1. \|f_{ds_nth} (f_{ds_mangoldt_complex}) n\| / n \text{ nat_powr } (b x))$$

$$= (\sum 'n \geq 1. \text{mangoldt_real } n / n \text{ nat_powr } (b x))$$

by (*subst norm_fds_mangoldt_complex*) (*rule refl*)

also have ... = - Re (logderiv zeta (b x))

proof -

have (($\lambda n. \text{mangoldt_real } n * n \text{ nat_powr } (-b x) * \cos (0 * \ln (\text{real } n))$)
has_sum Re (- deriv zeta (Complex (b x) 0) / zeta (Complex (b x) 0))) {1..}
by (intro sums_Re_logderiv_zeta) (use Hb in auto)

moreover have Complex (b x) 0 = b x by (rule complex_eqI) auto

moreover have Re (- deriv zeta (b x) / zeta (b x)) = - Re (logderiv zeta (b x))
unfolding logderiv_def by auto

ultimately have (($\lambda n. \text{mangoldt_real } n * n \text{ nat_powr } (-b x)$) has_sum
- Re (logderiv zeta (b x))) {1..} by auto

hence - Re (logderiv zeta (b x)) = ($\sum 'n \geq 1. \text{mangoldt_real } n * n \text{ nat_powr } (-b x)$)
by (intro has_sum_imp_has_subsum subsumI)

also have ... = ($\sum 'n \geq 1. \text{mangoldt_real } n / n \text{ nat_powr } (b x)$)

by (intro subsum_cong) (auto simp add: powr_minus_divide)

finally show ?thesis by auto

qed

also have ... $\leq |Re (\text{logderiv zeta } (b x))|$ by auto

also have ... $\leq \|\text{logderiv zeta } (b x)\|$ by (rule abs_Re_le_cmod)

also have ... $\leq 5 / 4 * (1 / (b x - 1))$

by (rule logderiv_zeta_bound) (use Hb in auto)

also have ... = B x unfolding b_def B_def by auto

finally show ?thesis .

qed

hence $\forall_F x \text{ in } at_top. 1 < b x \wedge b x \leq 23 / 20 \longrightarrow ?P x$ by auto

moreover have $\forall_F x \text{ in } at_top. b x \leq 23 / 20$ unfolding b_def by real_asymp

ultimately show ?thesis using Hb by eventually_elim auto

qed

ultimately have $\forall_F x \text{ in } at_top.$

$\| \text{sum_upto } (fds_nth (fds \text{ mangoldt_complex})) x - 1 / (2 * \pi * i)$
 $* \text{contour_integral } (P_2 x) (\lambda s. \text{eval_fds } (fds \text{ mangoldt_complex}) s * x \text{ powr } s / s) \| \leq R x$

unfolding R_def P2_def z2_def z3_def

by eventually_elim (rule perron_formula(2))

moreover have $\forall_F x \text{ in } at_top. \text{sum_upto } (fds_nth (fds \text{ mangoldt_complex})) x = \psi x$ for $x :: \text{real}$

unfolding primes_psi_def sum_upto_def by auto

moreover have

$\text{contour_integral } (P_2 x) (\lambda s. \text{eval_fds } (fds \text{ mangoldt_complex}) s * x \text{ powr } s / s)$

= $\text{contour_integral } (P_2 x) (\lambda s. - (x \text{ powr } s / s * \text{logderiv zeta } s))$

when $1 < b x$ for x

proof (rule contour_integral_eq, goal_cases)

case (1 s)

hence $Re s = b x$ by (rule Re_path_P2)

hence $\text{eval_fds } (fds \text{ mangoldt_complex}) s = - \text{deriv zeta } s / \text{zeta } s$

by (intro eval_fds_mangoldt) (use that in auto)

thus ?case unfolding logderiv_def by (auto simp add: field_simps)

qed

hence $\forall_F x \text{ in } at_top. 1 < b x \longrightarrow$

$\text{contour_integral } (P_2 x) (\lambda s. \text{eval_fds } (fds \text{ mangoldt_complex}) s * x \text{ powr } s / s)$
= $\text{contour_integral } (P_2 x) (\lambda s. - (x \text{ powr } s / s * \text{logderiv zeta } s))$

using Hb by (intro eventuallyI) blast

ultimately have $\forall_F x \text{ in } at_top.$

$\| \psi x - 1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (\lambda s. - (x \text{ powr } s / s * \text{logderiv zeta } s)) \| \leq R x$

using Hb by eventually_elim auto

thus ?thesis unfolding f_def

by eventually_elim (auto simp add: contour_integral_neg)

qed

lemma *estimation_perron_psi*:

$(\lambda x. \|\psi x + 1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x)\|) \in Rc$

proof –

have $(\lambda x. \|\psi x + 1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x)\|) \in O(R)$

by (*intro eventually_le_imp_bigo' perron_psi*)

moreover have $R \in Rc$ **by** (*rule estimation_R*)

ultimately show *?thesis unfolding Rc_def* **by** (*rule landau_o.big_trans*)

qed

theorem *prime_number_theorem*:

PNT_3 $(c / 2 - 2 * \varepsilon) (1 / 2) 0$

proof –

define r **where** $r x \equiv$

$\|\psi x + 1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x)\|$

$+ \|1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x) + x\|$ **for** x

have $\|\psi x - x\| \leq r x$ **for** x

proof –

have $\|\psi x - x\| = \|(\psi x :: \text{complex}) - x\|$

by (*fold dist_complex_def, simp add: dist_real_def*)

also have $\dots \leq \|\psi x - -1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x)\|$

$+ \|x - -1 / (2 * \pi * i) * \text{contour_integral } (P_2 x) (f x)\|$

by (*fold dist_complex_def, rule dist_triangle2*)

finally show *?thesis unfolding r_def* **by** (*simp add: add_ac*)

qed

hence $(\lambda x. \psi x - x) \in O(r)$ **by** (*rule le_imp_bigo*)

moreover have $r \in Rc$

unfolding *r_def Rc_def*

by (*intro sum_in_bigo, fold Rc_def*)

(*rule estimation_perron_psi, rule estimation_P2*)

ultimately show *?thesis unfolding PNT_3_def*

by (*subst Rc_eq_rem_est [symmetric], unfold Rc_def*)

(*rule landau_o.big_trans*)

qed

no_notation *primes_psi* (ψ)

end

unbundle *prime_counting_notation*

theorem *prime_number_theorem*:

shows $(\lambda x. \pi x - Li x) \in O(\lambda x. x * \exp(-1 / 3653 * (\ln x) \text{ powr } (1 / 2)))$

proof –

define $c :: \text{real}$ **where** $c \equiv 1 / 1826$

define $\varepsilon :: \text{real}$ **where** $\varepsilon \equiv 1 / 26681512$

interpret z : *prime_number_theorem* $c \varepsilon$

unfolding *c_def* *ε_def* **by** *standard (auto simp: C4_def)*

have *PNT_3* $(c / 2 - 2 * \varepsilon) (1 / 2) 0$ **by** (*rule z.prime_number_theorem*)

hence *PNT_1* $(c / 2 - 2 * \varepsilon) (1 / 2) 0$ **by** (*auto intro: PNT_3_imp_PNT_1*)

thus $(\lambda x. \pi x - Li x) \in O(\lambda x. x * \exp(-1 / 3653 * (\ln x) \text{ powr } (1 / 2)))$

unfolding *PNT_1_def rem_est_def c_def ε_def*

by (*rule landau_o.big.ev_eq_trans1, use ln_ln_asymp_pos in eventually_elim*)

(*auto intro: eventually_at_top_linorderI [of 1] simp: powr_half_sqrt*)

qed

```
hide_const (open) C3 C4 C5  
unbundle no_prime_counting_notation  
unbundle no_pnt_notation  
end
```