The Lambert W Function on the Reals

Manuel Eberl

March 17, 2025

Abstract

The Lambert W function is a multi-valued function defined as the inverse function of $x \mapsto xe^x$. Besides numerous applications in combinatorics, physics, and engineering, it also frequently occurs when solving equations containing both e^x and x, or both x and $\log x$.

This article provides a definition of the two real-valued branches $W_0(x)$ and $W_{-1}(x)$ and proves various properties such as basic identities and inequalities, monotonicity, differentiability, asymptotic expansions, and the MacLaurin series of $W_0(x)$ at x = 0.

Contents

1	The	Lambert W Function on the reals	2
	1.1	Properties of the function $x \mapsto xe^x$	2
	1.2	Definition	4
	1.3	Monotonicity properties	8
	1.4	Basic identities and bounds	15
	1.5	Limits, continuity, and differentiability	18
	1.6	Asymptotic expansion	24
	1.7	The MacLaurin series of $W_0(x)$ at $x = 0$	30

1 The Lambert *W* Function on the reals

```
theory Lambert-W

imports

Complex-Main

HOL-Library.FuncSet

HOL-Real-Asymp.Real-Asymp

begin
```

1.1 Properties of the function $x \mapsto xe^x$

lemma *exp-times-self-gt*: assumes $x \neq -1$ shows $x * exp \ x > -exp \ (-1::real)$ proof define f where $f = (\lambda x::real. x * exp x)$ define f' where $f' = (\lambda x :: real. (x + 1) * exp x)$ have (f has field - derivative f' x) (at x) for x by (auto simp: f-def f'-def introl: derivative-eq-intros simp: algebra-simps) define l r where l = min x (-1) and r = max x (-1)have $\exists z. z > l \land z < r \land fr - fl = (r - l) * f'z$ unfolding f-def f'-def l-def r-def using assms by (intro MVT2) (auto introl: derivative-eq-intros simp: algebra-simps) then obtain z where z: $z \in \{l < ... < r\}$ f r - f l = (r - l) * f' zby auto from z have f x = f (-1) + (x + 1) * f' zusing assms by (cases $x \ge -1$) (auto simp: l-def r-def max-def min-def algebra-simps) moreover have sqn ((x + 1) * f' z) = 1using z assmsby (cases x(-1)):: real rule: linorder-cases; cases z(-1):: real rule: linorder-cases) (auto simp: f'-def sgn-mult l-def r-def) hence (x + 1) * f' z > 0 using sgn-greater by fastforce ultimately show ?thesis by (simp add: f-def) qed **lemma** exp-times-self-ge: $x * exp \ x \ge -exp \ (-1::real)$ using exp-times-self-gt[of x] by (cases x = -1) auto lemma exp-times-self-strict-mono: assumes $x \ge -1 x < (y :: real)$ shows $x * exp \ x < y * exp \ y$ using assms(2)proof (rule DERIV-pos-imp-increasing-open) fix t assume t: x < t t < yhave $((\lambda x. x * exp x) has$ -real-derivative (t + 1) * exp t) (at t) **by** (*auto intro*!: *derivative-eq-intros simp*: *algebra-simps*)

moreover have $(t + 1) * exp \ t > 0$

using t assms by (intro mult-pos-pos) auto ultimately show $\exists y$. ((λa . a * exp a) has-real-derivative y) (at t) $\land 0 < y$ by blast**qed** (*auto intro*!: *continuous-intros*) **lemma** *exp-times-self-strict-antimono*: assumes $y \leq -1 x < (y :: real)$ shows $x * exp \ x > y * exp \ y$ proof have $-x * exp \ x < -y * exp \ y$ using assms(2)**proof** (*rule DERIV-pos-imp-increasing-open*) fix t assume t: x < t t < yhave $((\lambda x. -x * exp x) has-real-derivative (-(t + 1)) * exp t) (at t)$ **by** (*auto intro*!: *derivative-eq-intros simp*: *algebra-simps*) moreover have (-(t+1)) * exp t > 0using t assms by (intro mult-pos-pos) auto ultimately show $\exists y$. ((λa . -a * exp a) has-real-derivative y) (at t) $\land 0 < y$ by blast **qed** (*auto intro*!: *continuous-intros*) thus ?thesis by simp \mathbf{qed} **lemma** *exp-times-self-mono*: assumes $x \ge -1$ $x \le (y :: real)$ shows $x * exp \ x \le y * exp \ y$ using exp-times-self-strict-mono[of x y] assms by (cases x = y) auto **lemma** *exp-times-self-antimono*: assumes $y \leq -1 \ x \leq (y :: real)$ shows $x * exp \ x \ge y * exp \ y$ using exp-times-self-strict-antimono[of y x] assms by (cases x = y) auto **lemma** exp-times-self-inj: inj-on $(\lambda x::real. x * exp x) \{-1..\}$ proof fix x y :: real**assume** $x \in \{-1..\} y \in \{-1..\} x * exp x = y * exp y$ thus x = y**using** exp-times-self-strict-mono[of x y] exp-times-self-strict-mono[of y x] **by** (cases x y rule: linorder-cases) auto qed **lemma** exp-times-self-inj': inj-on $(\lambda x::real. x * exp x) \{..-1\}$ proof fix x y :: real**assume** $x \in \{..-1\}$ $y \in \{..-1\}$ $x * exp \ x = y * exp \ y$ thus x = y**using** exp-times-self-strict-antimono[of x y] exp-times-self-strict-antimono[of yx



Figure 1: The two real branches of the Lambert W function: W_0 (blue) and W_{-1} (red).

by (cases x y rule: linorder-cases) auto **qed**

1.2 Definition

The following are the two branches $W_0(x)$ and $W_{-1}(x)$ of the Lambert W function on the real numbers. These are the inverse functions of the function $x \mapsto xe^x$, i.e. we have $W(x)e^{W(x)} = x$ for both branches wherever they are defined. The two branches meet at the point $x = -\frac{1}{e}$.

 $W_0(x)$ is the principal branch, whose domain is $\left[-\frac{1}{e};\infty\right)$ and whose range is $\left[-1;\infty\right)$. $W_{-1}(x)$ has the domain $\left[-\frac{1}{e};0\right)$ and the range $\left(-\infty;-1\right]$. Figure 1 shows plots of these two branches for illustration.

definition Lambert-W :: real \Rightarrow real where Lambert-W $x = (if \ x < -exp(-1) \ then \ -1 \ else \ (THE \ w. \ w \ge -1 \ \land \ w \ exp \ w = x))$

definition Lambert-W':: real \Rightarrow real where

Lambert-W' $x = (if \ x \in \{-exp(-1)..<0\}$ then (THE w. $w \leq -1 \land w * exp \ w = x)$ else -1)

lemma Lambert-W-ex1:

assumes $(x::real) \ge -exp(-1)$ shows $\exists ! w. w \ge -1 \land w * exp w = x$ **proof** (*rule ex-ex11*) have filterlim (λw ::real. w * exp w) at-top at-top **by** real-asymp hence eventually ($\lambda w. w * exp \ w \ge x$) at-top **by** (*auto simp: filterlim-at-top*) hence eventually ($\lambda w. w \ge 0 \land w * exp \ w \ge x$) at-top **by** (*intro eventually-conj eventually-ge-at-top*) then obtain w' where w': $w' * exp \ w' \ge x \ w' \ge 0$ **by** (*auto simp: eventually-at-top-linorder*) from w' assms have $\exists w. -1 \leq w \land w \leq w' \land w * exp w = x$ **by** (*intro* IVT' continuous-intros) auto thus $\exists w. w \geq -1 \land w * exp w = x$ by blast \mathbf{next} fix w w' :: realassume ww': $w \ge -1 \land w * exp \ w = x \ w' \ge -1 \land w' * exp \ w' = x$ hence w * exp w = w' * exp w' by simp thus w = w'using exp-times-self-strict-mono[of w w'] exp-times-self-strict-mono[of w' w] ww'by (cases w w' rule: linorder-cases) auto qed **lemma** Lambert-W'-ex1: assumes $(x::real) \in \{-exp (-1)..<0\}$ shows $\exists ! w. w \leq -1 \land w * exp w = x$ **proof** (*rule ex-ex11*) have eventually $(\lambda w. x \leq w * exp w)$ at-bot using assms by real-asymp hence eventually ($\lambda w. w \leq -1 \land w * exp \ w \geq x$) at-bot **by** (*intro eventually-conj eventually-le-at-bot*) then obtain w' where w': $w' * exp \ w' \ge x \ w' \le -1$ **by** (*auto simp: eventually-at-bot-linorder*) from w' assms have $\exists w. w' \leq w \land w \leq -1 \land w * exp w = x$ by (intro IVT2' continuous-intros) auto thus $\exists w. w \leq -1 \land w * exp w = x$ by blast \mathbf{next} fix w w' :: realassume ww': $w \leq -1 \land w * exp \ w = x \ w' \leq -1 \land w' * exp \ w' = x$ hence w * exp w = w' * exp w' by simp thus w = w'using exp-times-self-strict-antimono[of w w'] exp-times-self-strict-antimono[of w' w w w'by (cases w w' rule: linorder-cases) auto ged

lemma Lambert-W-times-exp-self:

assumes $x \ge -exp(-1)$ **shows** Lambert-W x * exp (Lambert-W x) = xusing the I'[OF Lambert-W-ex1[OF assms]] assms by (auto simp: Lambert-W-def) **lemma** Lambert-W-times-exp-self': assumes $x \ge -exp(-1)$ **shows** exp (Lambert-W x) * Lambert-W x = x**using** Lambert-W-times-exp-self [of x] assms by (simp add: mult-ac) **lemma** Lambert-W'-times-exp-self: assumes $x \in \{-exp \ (-1)..<\theta\}$ **shows** Lambert-W' x * exp (Lambert-W' x) = x using the I'[OF Lambert-W'-ex1[OF assms]] assms by (auto simp: Lambert-W'-def) **lemma** Lambert-W'-times-exp-self': assumes $x \in \{-exp \ (-1)..<0\}$ **shows** exp (Lambert-W' x) * Lambert-W' x = xusing Lambert-W'-times-exp-self [of x] assms by (simp add: mult-ac) **lemma** Lambert-W-ge: Lambert-W $x \ge -1$ using the I'[OF Lambert-W-ex1 [of x]] by (auto simp: Lambert-W-def) lemma Lambert-W'-le: Lambert-W' $x \leq -1$ using the I'[OF Lambert-W'-ex1[of x]] by (auto simp: Lambert-W'-def) **lemma** Lambert-W-eqI: assumes $w \ge -1 \ w * exp \ w = x$ **shows** Lambert-W x = wproof from assms exp-times-self-ge[of w] have $x \ge -exp(-1)$ by (cases $x \ge -exp(-1)$) auto from Lambert-W-ex1 [OF this] Lambert-W-times-exp-self [OF this] Lambert-W-ge [of x] assms show ?thesis by metis qed **lemma** Lambert-W'-eqI: assumes $w < -1 \ w * exp \ w = x$ **shows** Lambert-W' x = wproof from assms exp-times-self-ge[of w] have $x \ge -exp(-1)$ by (cases $x \ge -exp(-1)$) auto moreover from assms have $w * exp \ w < 0$ by (intro mult-neg-pos) auto ultimately have $x \in \{-exp \ (-1)..<\theta\}$ using assms by auto

 $\label{eq:constraint} \begin{array}{l} \mbox{from } Lambert\mbox{-}W'\mbox{-}exp\mbox{-}self[OF\mbox{ this}(1)]\mbox{ Lambert\mbox{-}W'\mbox{-}le\mbox{ assms}} \end{array}$

show ?thesis by metis qed $W_0(x)$ and $W_{-1}(x)$ together fully cover all solutions of $we^w = x$: **lemma** *exp-times-self-eqD*: assumes $w * exp \ w = x$ shows $x \ge -exp(-1)$ and $w = Lambert \cdot W x \lor x < 0 \land w = Lambert \cdot W' x$ proof – from assms show $x \ge -exp(-1)$ using exp-times-self-ge[of w] by auto show $w = Lambert W x \lor x < 0 \land w = Lambert W' x$ **proof** (cases $w \ge -1$) case True hence Lambert-W x = wusing assms by (intro Lambert-W-eqI) auto thus ?thesis by auto \mathbf{next} case False from False have $w * exp \ w < 0$ by (intro mult-neg-pos) auto from False have Lambert-W' x = wusing assms by (intro Lambert-W'-eqI) auto thus ?thesis using assms $\langle w \ast exp \ w < 0 \rangle$ by auto \mathbf{qed} qed **theorem** *exp-times-self-eq-iff*: $w * exp \ w = x \longleftrightarrow x \ge -exp \ (-1) \land (w = Lambert-W \ x \lor x < 0 \land w = x$ Lambert-W'(x)using exp-times-self-eqD[of w x] by (auto simp: Lambert-W-times-exp-self Lambert-W'-times-exp-self) **lemma** Lambert-W-exp-times-self [simp]: $x \ge -1 \implies$ Lambert-W (x * exp x) = x**by** (rule Lambert-W-eqI) auto **lemma** Lambert-W-exp-times-self' [simp]: $x \ge -1 \implies$ Lambert-W (exp x * x) = xby (rule Lambert-W-eqI) auto **lemma** Lambert-W'-exp-times-self [simp]: $x \leq -1 \implies$ Lambert-W' (x * exp x) = xby (rule Lambert-W'-eqI) auto **lemma** Lambert-W'-exp-times-self' [simp]: $x \leq -1 \implies$ Lambert-W' (exp x * x) = xby (rule Lambert-W'-eqI) auto **lemma** Lambert-W-times-ln-self:

assumes $x \ge exp(-1)$ shows Lambert-W $(x * \ln x) = \ln x$ proof – have 0 < exp(-1 :: real)by simp also note $\langle \dots \le x \rangle$ finally have x > 0. from assms have $\ln (exp(-1)) \le \ln x$ using $\langle x > 0 \rangle$ by (subst ln-le-cancel-iff) auto hence Lambert-W (exp (ln x) * ln x) = ln x by (subst Lambert-W-exp-times-self') auto thus ?thesis using $\langle x > 0 \rangle$ by simp qed

lemma Lambert-W-times-ln-self': **assumes** $x \ge exp(-1)$ **shows** Lambert-W (ln x * x) = ln x**using** Lambert-W-times-ln-self[OF assms] **by** (simp add: mult.commute)

lemma Lambert-W-eq-minus-exp-minus1 [simp]: Lambert-W (-exp(-1)) = -1by (rule Lambert-W-eqI) auto

lemma Lambert-W'-eq-minus-exp-minus1 [simp]: Lambert-W' (-exp(-1)) = -1by (rule Lambert-W'-eqI) auto

```
lemma Lambert-W-0 [simp]: Lambert-W 0 = 0
by (rule Lambert-W-eqI) auto
```

1.3 Monotonicity properties

 $\begin{array}{l} \textbf{lemma Lambert-W-strict-mono:}\\ \textbf{assumes } x \geq -exp(-1) \ x < y\\ \textbf{shows Lambert-W } x < Lambert-W \ y\\ \textbf{proof } (rule \ ccontr)\\ \textbf{assume } \neg (Lambert-W \ x < Lambert-W \ y)\\ \textbf{hence Lambert-W } x \ * exp \ (Lambert-W \ x) \geq Lambert-W \ y \ * exp \ (Lambert-W \ y)\\ \textbf{by } (intro \ exp-times-self-mono) \ (auto \ simp: \ Lambert-W-ge)\\ \textbf{hence } x \geq y\\ \textbf{using } assms \ \textbf{by } (simp \ add: \ Lambert-W-times-exp-self)\\ \textbf{with } assms \ \textbf{show } False \ \textbf{by } simp\\ \textbf{qed} \end{array}$

lemma Lambert-W-mono: **assumes** $x \ge -exp(-1)$ $x \le y$ **shows** Lambert-W $x \le Lambert-W y$ **using** Lambert-W-strict-mono[of x y] assms by (cases x = y) auto

lemma Lambert-W-eq-iff [simp]:

$$x \ge -exp(-1) \Longrightarrow y \ge -exp(-1) \Longrightarrow$$
 Lambert-W $x =$ Lambert-W $y \longleftrightarrow x = y$

using Lambert-W-strict-mono[of x y] Lambert-W-strict-mono[of y x] **by** (cases x y rule: linorder-cases) auto

lemma Lambert-W-le-iff [simp]: $x \ge -exp(-1) \Longrightarrow y \ge -exp(-1) \Longrightarrow Lambert W x \le Lambert W y \longleftrightarrow x \le y$ **using** Lambert-W-strict-mono[of x y] Lambert-W-strict-mono[of y x] by (cases x y rule: linorder-cases) auto **lemma** Lambert-W-less-iff [simp]: $x \ge -exp(-1) \Longrightarrow y \ge -exp(-1) \Longrightarrow Lambert-W x < Lambert-W y \longleftrightarrow x < y$ **using** Lambert-W-strict-mono[of x y] Lambert-W-strict-mono[of y x] by (cases x y rule: linorder-cases) auto lemma Lambert-W-le-minus-one: assumes x < -exp(-1)shows Lambert-W x = -1**proof** (cases x = -exp(-1)) case False thus ?thesis using assms by (auto simp: Lambert-W-def) qed auto **lemma** Lambert-W-pos-iff [simp]: Lambert-W $x > 0 \iff x > 0$ **proof** (cases $x \ge -exp(-1)$) case True thus ?thesis using Lambert-W-less-iff of 0 x by (simp del: Lambert-W-less-iff) next case False hence x < -exp(-1) by *auto* also have $\ldots \leq \theta$ by simp finally show ?thesis using False **by** (*auto simp: Lambert-W-le-minus-one*) qed **lemma** Lambert-W-eq-0-iff [simp]: Lambert-W $x = 0 \iff x = 0$ using Lambert-W-eq-iff[of x 0] by (cases $x \ge -exp(-1)$) (auto simp: Lambert-W-le-minus-one simp del: Lambert-W-eq-iff) **lemma** Lambert-W-nonneg-iff [simp]: Lambert-W $x \ge 0 \iff x \ge 0$ using Lambert-W-pos-iff [of x]by (cases x = 0) (auto simp del: Lambert-W-pos-iff) **lemma** Lambert-W-neg-iff [simp]: Lambert-W $x < 0 \leftrightarrow x < 0$ using Lambert-W-nonneg-iff [of x] by (auto simp del: Lambert-W-nonneg-iff) **lemma** Lambert-W-nonpos-iff [simp]: Lambert-W $x \leq 0 \iff x \leq 0$

using Lambert-W-pos-iff [of x] by (auto simp del: Lambert-W-pos-iff)

```
lemma Lambert-W-geI:
 assumes y * exp \ y \le x
 shows Lambert-W x \ge y
proof (cases y \ge -1)
 case False
 hence y \leq -1 by simp
 also have -1 \leq Lambert W x by (rule Lambert W-ge)
 finally show ?thesis .
\mathbf{next}
 case True
 have Lambert-W x \ge Lambert-W (y * exp y)
   using assms exp-times-self-ge[of y] by (intro Lambert-W-mono) auto
 thus ?thesis using assms True by simp
qed
lemma Lambert-W-qtI:
 assumes y * exp \ y < x
 shows Lambert-W x > y
proof (cases y \ge -1)
 case False
 hence y < -1 by simp
 also have -1 \leq Lambert-W x by (rule Lambert-W-ge)
 finally show ?thesis .
\mathbf{next}
 case True
 have Lambert-W x > Lambert-W (y * exp y)
   using assms exp-times-self-ge[of y] by (intro Lambert-W-strict-mono) auto
 thus ?thesis using assms True by simp
qed
lemma Lambert-W-leI:
 assumes y * exp \ y \ge x \ y \ge -1 \ x \ge -exp \ (-1)
 shows Lambert-W x \leq y
proof -
 have Lambert-W x \leq Lambert-W (y * exp y)
   using assms exp-times-self-ge[of y] by (intro Lambert-W-mono) auto
 thus ?thesis using assms by simp
qed
lemma Lambert-W-lessI:
 assumes y * exp \ y > x \ y \ge -1 \ x \ge -exp \ (-1)
 shows Lambert-W x < y
proof -
 have Lambert-W x < Lambert-W (y * exp y)
   using assms exp-times-self-ge[of y] by (intro Lambert-W-strict-mono) auto
 thus ?thesis using assms by simp
qed
```

lemma Lambert-W'-strict-antimono: assumes $-exp(-1) \leq x x < y y < 0$ **shows** Lambert-W' x > Lambert-W' y**proof** (*rule ccontr*) assume \neg (Lambert-W' x > Lambert-W' y) hence Lambert-W' x * exp (Lambert-W' x) \geq Lambert-W' y * exp (Lambert-W' y)using assms by (intro exp-times-self-antimono Lambert-W'-le) auto hence $x \ge y$ using assms by (simp add: Lambert-W'-times-exp-self) with assms show False by simp qed **lemma** Lambert-W'-antimono: assumes $x \ge -exp(-1)$ $x \le y y < 0$ **shows** Lambert- $W' x \ge Lambert-W' y$ using Lambert-W'-strict-antimono[of x y] assms by (cases x = y) auto **lemma** Lambert-W'-eq-iff [simp]: $x \in \{-exp(-1)..<0\} \Longrightarrow y \in \{-exp(-1)..<0\} \Longrightarrow Lambert-W' x = Lambert-W'$ $y \longleftrightarrow x = y$ using Lambert-W'-strict-antimono[of x y] Lambert-W'-strict-antimono[of y x] **by** (cases x y rule: linorder-cases) auto **lemma** Lambert-W'-le-iff [simp]: $x \in \{-exp(-1)..<0\} \Longrightarrow y \in \{-exp(-1)..<0\} \Longrightarrow Lambert-W' x \leq Lambert$ $y \longleftrightarrow x \ge y$ **using** Lambert-W'-strict-antimono[of x y] Lambert-W'-strict-antimono[of y x] by (cases x y rule: linorder-cases) auto **lemma** Lambert-W'-less-iff [simp]: $x \in \{-exp(-1)..<0\} \Longrightarrow y \in \{-exp(-1)..<0\} \Longrightarrow Lambert-W' x < Lambert$ $y \longleftrightarrow x > y$ using Lambert-W'-strict-antimono[of x y] Lambert-W'-strict-antimono[of y x] **by** (cases x y rule: linorder-cases) auto **lemma** Lambert-W'-le-minus-one: assumes $x \leq -exp(-1)$ **shows** Lambert-W' x = -1**proof** (cases x = -exp(-1)) case False thus ?thesis using assms **by** (auto simp: Lambert-W'-def) qed auto **lemma** Lambert-W'-ge-zero: $x \ge 0 \implies$ Lambert-W' x = -1

by (simp add: Lambert-W'-def)

```
lemma Lambert-W'-neg: Lambert-W' x < 0
 by (rule le-less-trans [OF Lambert-W'-le]) auto
lemma Lambert-W'-nz [simp]: Lambert-W' x \neq 0
 using Lambert-W'-neg[of x] by simp
lemma Lambert-W'-geI:
 assumes y * exp \ y \ge x \ y \le -1 \ x \ge -exp(-1)
 shows Lambert-W' x \ge y
proof -
 from assms have y * exp \ y < 0
   by (intro mult-neg-pos) auto
 hence Lambert-W' x \ge Lambert-W' (y * exp y)
   using assms exp-times-self-ge[of y] by (intro Lambert-W'-antimono) auto
 thus ?thesis using assms by simp
qed
lemma Lambert-W'-gtI:
 assumes y * exp \ y > x \ y \le -1 \ x \ge -exp(-1)
 shows Lambert-W' x \ge y
proof -
 from assms have y * exp \ y < 0
   by (intro mult-neg-pos) auto
 hence Lambert-W' x > Lambert-W' (y * exp y)
  using assms exp-times-self-ge[of y] by (intro Lambert-W'-strict-antimono) auto
 thus ?thesis using assms by simp
qed
lemma Lambert-W'-leI:
 assumes y * exp \ y \le x \ x < 0
 shows Lambert-W' x \leq y
proof (cases y \leq -1)
 case True
 have Lambert-W' x \leq Lambert-W' (y * exp y)
   using assms exp-times-self-qe[of y] by (intro Lambert-W'-antimono) auto
 thus ?thesis using assms True by simp
\mathbf{next}
 case False
 have Lambert-W' x \leq -1
  by (rule Lambert-W'-le)
 also have \ldots < y
   using False by simp
 finally show ?thesis by simp
qed
lemma Lambert-W'-lessI:
 assumes y * exp \ y < x \ x < 0
 shows Lambert-W' x < y
```

proof (cases $y \le -1$) case True have Lambert-W' x < Lambert-W' (y * exp y)using assms exp-times-self-ge[of y] by (intro Lambert-W'-strict-antimono) auto thus ?thesis using assms True by simp next case False have Lambert-W' $x \le -1$ by (rule Lambert-W'-le) also have ... < y using False by simp finally show ?thesis by simp qed

```
lemma bij-betw-exp-times-self-atLeastAtMost:
 fixes a b :: real
 assumes a \ge -1 a \le b
 shows bij-betw (\lambda x. \ x * exp \ x) \{a..b\} \{a * exp \ a..b * exp \ b\}
 unfolding bij-betw-def
proof
 show inj-on (\lambda x. x * exp x) \{a..b\}
   by (rule inj-on-subset[OF exp-times-self-inj]) (use assms in auto)
\mathbf{next}
 show (\lambda x. \ x * exp \ x) ' \{a..b\} = \{a * exp \ a..b * exp \ b\}
 proof safe
   fix x assume x \in \{a..b\}
   thus x * exp \ x \in \{a * exp \ a..b * exp \ b\}
     using assms by (auto intro!: exp-times-self-mono)
 \mathbf{next}
   fix x assume x: x \in \{a * exp \ a..b * exp \ b\}
   have (-1) * exp(-1) \leq a * exp a
     using assms by (intro exp-times-self-mono) auto
   also have \ldots \leq x using x by simp
   finally have x \ge -exp(-1) by simp
   have Lambert-W x \in \{a..b\}
       using x \langle x \geq -exp (-1) \rangle assms by (auto introl: Lambert-W-geI Lam-
bert-W-leI)
   moreover have Lambert-W x * exp (Lambert-W x) = x
     using \langle x \geq -exp(-1) \rangle by (simp add: Lambert-W-times-exp-self)
   ultimately show x \in (\lambda x. \ x * exp \ x) ' {a..b}
     unfolding image-iff by metis
 qed
qed
lemma bij-betw-exp-times-self-atLeastAtMost':
 fixes a b :: real
 assumes a \leq b \ b \leq -1
```

```
shows bij-betw (\lambda x. \ x * exp \ x) \{a..b\} \{b * exp \ b..a * exp \ a\}
  unfolding bij-betw-def
proof
  show inj-on (\lambda x. x * exp x) \{a..b\}
   by (rule inj-on-subset[OF exp-times-self-inj']) (use assms in auto)
\mathbf{next}
  show (\lambda x. \ x * exp \ x) ' \{a..b\} = \{b * exp \ b..a * exp \ a\}
 proof safe
   fix x assume x \in \{a..b\}
   thus x * exp \ x \in \{b * exp \ b..a * exp \ a\}
     using assms by (auto introl: exp-times-self-antimono)
 \mathbf{next}
   fix x assume x: x \in \{b * exp \ b..a * exp \ a\}
   from assms have a * exp \ a < 0
     by (intro mult-neq-pos) auto
   with x have x < \theta by auto
   have (-1) * exp (-1) \le b * exp b
     using assms by (intro exp-times-self-antimono) auto
   also have \ldots \leq x using x by simp
   finally have x \ge -exp(-1) by simp
   have Lambert-W' x \in \{a..b\}
     using x \langle x \geq -exp(-1) \rangle \langle x < 0 \rangle assms
     by (auto introl: Lambert-W'-geI Lambert-W'-leI)
   moreover have Lambert-W' x * exp (Lambert-W' x) = x
     using \langle x \geq -exp(-1) \rangle \langle x < 0 \rangle by (auto simp: Lambert-W'-times-exp-self)
   ultimately show x \in (\lambda x. \ x * exp \ x) ' {a..b}
     unfolding image-iff by metis
 qed
qed
lemma bij-betw-exp-times-self-atLeast:
 fixes a :: real
 assumes a \geq -1
 shows bij-betw (\lambda x. \ x * exp \ x) \{a..\} \{a * exp \ a..\}
 unfolding bij-betw-def
proof
 show inj-on (\lambda x. x * exp x) \{a..\}
   by (rule inj-on-subset[OF exp-times-self-inj]) (use assms in auto)
\mathbf{next}
 show (\lambda x. \ x * exp \ x) ' \{a..\} = \{a * exp \ a..\}
 proof safe
   fix x assume x \ge a
   thus x * exp \ x \ge a * exp \ a
     using assms by (auto intro!: exp-times-self-mono)
  \mathbf{next}
   fix x assume x: x \ge a * exp a
   have (-1) * exp (-1) \le a * exp a
     using assms by (intro exp-times-self-mono) auto
```

also have $\ldots \leq x$ using x by simp finally have $x \geq -exp(-1)$ by simp have Lambert-W $x \in \{a..\}$ using $x \langle x \geq -exp(-1) \rangle$ assms by (auto introl: Lambert-W-geI Lambert-W-leI) moreover have Lambert-W x * exp (Lambert-W x) = x using $\langle x \geq -exp(-1) \rangle$ by (simp add: Lambert-W-times-exp-self) ultimately show $x \in (\lambda x. \ x * exp \ x)$ ' {a..} unfolding image-iff by metis qed

1.4 Basic identities and bounds

```
lemma Lambert-W-2-ln-2 [simp]: Lambert-W (2 * ln 2) = ln 2
proof -
 have -1 \leq (0 :: real)
   by simp
 also have \ldots \leq \ln 2
   by simp
 finally have -1 \leq (\ln 2 :: real).
 thus ?thesis
   by (intro Lambert-W-eqI) auto
qed
lemma Lambert-W-exp-1 [simp]: Lambert-W (exp 1) = 1
 by (rule Lambert-W-eqI) auto
lemma Lambert-W-neq-ln-over-self:
 assumes x \in \{exp \ (-1)..exp \ 1\}
 shows Lambert-W (-\ln x / x) = -\ln x
proof –
 have \theta < (exp(-1) :: real)
   by simp
 also have \ldots \leq x
   using assms by simp
 finally have x > 0.
 from \langle x > 0 \rangle assms have \ln x \leq \ln (exp \ 1)
   by (subst ln-le-cancel-iff) auto
 also have ln (exp 1) = (1 :: real)
   by simp
 finally have ln \ x \le 1.
 show ?thesis
   using assms \langle x > 0 \rangle \langle \ln x \leq 1 \rangle
   by (intro Lambert-W-eqI) (auto simp: exp-minus field-simps)
qed
```

lemma Lambert-W'-neg-ln-over-self:

assumes $x \ge exp \ 1$ **shows** Lambert- $W'(-\ln x / x) = -\ln x$ **proof** (*rule* Lambert-W'-eqI) have $\theta < (exp \ 1 :: real)$ by simp also have $\ldots \leq x$ by fact finally have x > 0. from assms $\langle x > 0 \rangle$ have $\ln x \ge \ln (exp \ 1)$ by (subst ln-le-cancel-iff) auto thus $-\ln x \leq -1$ by simpshow $-\ln x * exp(-\ln x) = -\ln x / x$ using $\langle x > 0 \rangle$ by (simp add: field-simps exp-minus) qed **lemma** exp-Lambert-W: $x \ge -exp(-1) \Longrightarrow x \ne 0 \Longrightarrow exp(Lambert-W x) = x$ / Lambert-W xusing Lambert-W-times-exp-self [of x] by (auto simp add: divide-simps mult-ac) **lemma** exp-Lambert-W': $x \in \{-exp \ (-1)..<0\} \implies exp \ (Lambert-W' \ x) = x \ /$ Lambert-W' xusing Lambert-W'-times-exp-self [of x] by (auto simp add: divide-simps mult-ac) **lemma** *ln-Lambert-W*: assumes $x > \theta$ shows ln (Lambert-W x) = ln x - Lambert-W xproof – have $-exp(-1) \leq (0 :: real)$ by simp also have $\ldots < x$ by fact finally have x: x > -exp(-1). have exp (ln (Lambert-W x)) = exp (ln x - Lambert-W x)using assms x by (subst exp-diff) (auto simp: exp-Lambert-W) thus ?thesis by (subst (asm) exp-inj-iff) qed lemma *ln-minus-Lambert-W'*: assumes $x \in \{-exp \ (-1)..<0\}$ **shows** ln (-Lambert-W'x) = ln (-x) - Lambert-W'xproof – have exp (ln (-x) - Lambert W' x) = -Lambert W' xusing assms by (simp add: exp-diff exp-Lambert-W') also have $\ldots = exp (ln (-Lambert - W' x))$ using Lambert-W'-neg[of x] by simp finally show ?thesis by simp qed

lemma Lambert-W-plus-Lambert-W-eq:

```
assumes x > 0 y > 0
 shows Lambert-W x + Lambert-W y = Lambert-W (x * y * (1 / Lambert-W))
x + 1 / Lambert W y))
proof (rule sym, rule Lambert-W-eqI)
 have x > -exp(-1) \ y > -exp(-1)
   by (rule less-trans[OF - assms(1)] less-trans[OF - assms(2)], simp)+
 with assms show (Lambert-W x + Lambert-W y) * exp (Lambert-W x + Lam-
bert - W y) =
                x * y * (1 / Lambert-W x + 1 / Lambert-W y)
   by (auto simp: field-simps exp-add exp-Lambert-W)
 have -1 \leq (0 :: real)
   by simp
 also from assms have \ldots \leq Lambert \cdot W x + Lambert \cdot W y
   by (intro add-nonneg-nonneg) auto
 finally show \ldots \ge -1.
qed
lemma Lambert-W'-plus-Lambert-W'-eq:
 assumes x \in \{-exp(-1)..<0\} y \in \{-exp(-1)..<0\}
 shows Lambert-W' x + Lambert-W' y = Lambert-W' (x * y * (1 / Lambert-W')
x + 1 / Lambert W' y)
proof (rule sym, rule Lambert-W'-eqI)
  from assms show (Lambert-W' x + Lambert-W' y) * exp (Lambert-W' x +
Lambert-W'(y) =
                x * y * (1 / Lambert-W' x + 1 / Lambert-W' y)
   by (auto simp: field-simps exp-add exp-Lambert-W')
 have Lambert-W' x + Lambert-W' y \leq -1 + -1
   by (intro add-mono Lambert-W'-le)
 also have \ldots \leq -1 by simp
 finally show Lambert-W' x + Lambert-W' y \leq -1.
qed
lemma Lambert-W-gt-ln-minus-ln-ln:
 assumes x > exp 1
 shows Lambert-W x > \ln x - \ln (\ln x)
proof (rule Lambert-W-qtI)
 have x > 1
   by (rule less-trans[OF - assms]) auto
 have ln x > ln (exp 1)
   by (subst ln-less-cancel-iff) (use \langle x > 1 \rangle assms in auto)
 thus (\ln x - \ln (\ln x)) * exp (\ln x - \ln (\ln x)) < x
   using assms \langle x > 1 \rangle by (simp add: exp-diff field-simps)
qed
lemma Lambert-W-less-ln:
 assumes x > exp 1
 shows Lambert-W x < \ln x
proof (rule Lambert-W-lessI)
 have x > \theta
```

by (rule less-trans[OF - assms]) auto have $\ln x > \ln (exp \ 1)$ by (subst ln-less-cancel-iff) (use $\langle x > 0 \rangle$ assms in auto) thus $x < \ln x * exp (\ln x)$ using $\langle x > 0 \rangle$ by simp show $\ln x \ge -1$ by (rule less-imp-le[OF le-less-trans[OF - $\langle \ln x > - \rangle$]]) auto show $x \ge -exp (-1)$ by (rule less-imp-le[OF le-less-trans[OF - $\langle x > 0 \rangle$]]) auto qed

1.5 Limits, continuity, and differentiability

```
lemma filterlim-Lambert-W-at-top [tendsto-intros]: filterlim Lambert-W at-top at-top
 unfolding filterlim-at-top
proof
 fix C :: real
 have eventually (\lambda x. x > C * exp C) at-top
   by (rule eventually-ge-at-top)
  thus eventually (\lambda x. Lambert-W x \ge C) at-top
 proof eventually-elim
   case (elim x)
   thus ?case
     by (intro Lambert-W-geI) auto
  qed
qed
lemma filterlim-Lambert-W-at-left-0 [tendsto-intros]:
 filterlim Lambert-W' at-bot (at-left 0)
 unfolding filterlim-at-bot
proof
 \mathbf{fix} \ C :: \ real
 define C' where C' = min \ C \ (-1)
 have C' < \theta \ C' \leq C
   by (simp-all \ add: \ C'-def)
 have C' * exp C' < \theta
   using \langle C' < \theta \rangle by (intro mult-neg-pos) auto
 hence eventually (\lambda x. x \geq C' * exp C') (at-left 0)
   by real-asymp
  moreover have eventually (\lambda x::real. x < 0) (at-left 0)
   by real-asymp
  ultimately show eventually (\lambda x. Lambert-W' x \leq C) (at-left \theta)
  proof eventually-elim
   case (elim x)
   hence Lambert-W' x \leq C'
     by (intro Lambert-W'-leI) auto
   also have \ldots \leq C by fact
   finally show ?case .
  qed
```

qed

lemma continuous-on-Lambert-W [continuous-intros]: continuous-on $\{-exp(-1),.\}$ Lambert-Wproof have *: continuous-on $\{-exp (-1).b * exp b\}$ Lambert-W if $b \ge 0$ for b proof – have continuous-on $((\lambda x. x * exp x) ` \{-1..b\})$ Lambert-W by (rule continuous-on-inv) (auto introl: continuous-intros) also have $(\lambda x. x * exp x)$ ' $\{-1..b\} = \{-exp (-1)..b * exp b\}$ using *bij-betw-exp-times-self-atLeastAtMost*[of -1 b] $\langle b \geq 0 \rangle$ **by** (*simp add: bij-betw-def*) finally show ?thesis . qed have continuous (at x) Lambert-W if x > 0 for x proof – have x: -exp(-1) < x**by** (rule less-le-trans[OF - that]) auto define b where b = Lambert W x + 1have $b \ge 0$ **using** Lambert-W-ge[of x] **by** (simp add: b-def) have x = Lambert W x * exp (Lambert W x)using that x by (subst Lambert-W-times-exp-self) auto also have $\ldots < b * exp b$ by (intro exp-times-self-strict-mono) (auto simp: b-def Lambert-W-ge) finally have $b * exp \ b > x$. have continuous-on $\{-exp(-1) < ... < b * exp b\}$ Lambert-W by (rule continuous-on-subset[OF *[of b]]) (use $\langle b \geq 0 \rangle$ in auto) moreover have $x \in \{-exp(-1) < .. < b * exp b\}$ using $\langle b * exp \ b > x \rangle x$ by auto ultimately show continuous (at x) Lambert-W by (subst (asm) continuous-on-eq-continuous-at) auto qed hence continuous-on $\{0..\}$ Lambert-W by (intro continuous-at-imp-continuous-on) auto moreover have continuous-on $\{-exp (-1)..0\}$ Lambert-W using $*[of \ 0]$ by simp ultimately have continuous-on $(\{-exp \ (-1)..0\} \cup \{0..\})$ Lambert-W by (intro continuous-on-closed-Un) auto also have $\{-exp \ (-1)..0\} \cup \{0..\} = \{-exp \ (-1::real)..\}$ using order.trans[of -exp(-1)::real 0] by auto finally show ?thesis . qed

lemma continuous-on-Lambert-W-alt [continuous-intros]: **assumes** continuous-on $A f \land x. x \in A \implies f x \ge -exp (-1)$ **shows** continuous-on $A (\lambda x. Lambert-W (f x))$ using continuous-on-compose 2[OF continuous-on-Lambert- W assms(1)] assms by auto

lemma continuous-on-Lambert-W' [continuous-intros]: continuous-on $\{-exp(-1), <0\}$ Lambert-W' proof have *: continuous-on $\{-exp (-1), -b * exp (-b)\}$ Lambert-W' if $b \ge 1$ for b proof – have continuous-on $((\lambda x. x * exp x) ` \{-b..-1\})$ Lambert-W' by (intro continuous-on-inv ballI) (auto intro!: continuous-intros) also have $(\lambda x. \ x * exp \ x) \ (-b..-1) = \{-exp \ (-1)..-b * exp \ (-b)\}$ using bij-betw-exp-times-self-atLeastAtMost' [of -b - 1] that **by** (*simp add: bij-betw-def*) finally show ?thesis . qed have continuous (at x) Lambert-W' if x > -exp(-1) x < 0 for x proof define b where b = Lambert W x + 1have eventually $(\lambda b. -b * exp (-b) > x)$ at-top using that by real-asymp hence eventually ($\lambda b. b \ge 1 \land -b * exp(-b) > x$) at-top **by** (*intro eventually-conj eventually-ge-at-top*) then obtain b where b: $b \ge 1 - b * exp(-b) > x$ **by** (*auto simp: eventually-at-top-linorder*) have continuous-on $\{-exp(-1) < .. < -b * exp(-b)\}$ Lambert-W' by (rule continuous-on-subset[OF *[of b]]) (use $\langle b \geq 1 \rangle$ in auto) moreover have $x \in \{-exp(-1) < .. < -b * exp(-b)\}$ using b that by auto ultimately show continuous (at x) Lambert-W' by (subst (asm) continuous-on-eq-continuous-at) auto \mathbf{qed} hence **: continuous-on $\{-exp (-1) < ... < 0\}$ Lambert-W' by (intro continuous-at-imp-continuous-on) auto show ?thesis unfolding continuous-on-def proof fix x :: real assume $x: x \in \{-exp(-1)..<0\}$ show (Lambert-W' \longrightarrow Lambert-W' x) (at x within $\{-exp(-1)..<0\}$) **proof** (cases x = -exp(-1)) case False **hence** is Cont Lambert-W' xusing $x \ast by$ (auto simp: continuous-on-eq-continuous-at) thus ?thesis using continuous-at filterlim-within-subset by blast next case True

define a :: real where a = -2 * exp(-2)have a: a > -exp(-1)using exp-times-self-strict-antimono[of -1 -2] by (auto simp: a-def) from True have $x \in \{-exp \ (-1).. < a\}$ using a by (auto simp: a-def) have continuous-on $\{-exp (-1)..< a\}$ Lambert-W' unfolding a-def by (rule continuous-on-subset[OF * [of 2]]) auto hence (Lambert-W' \longrightarrow Lambert-W' x) (at x within $\{-exp (-1)..<a\}$) using $\langle x \in \{-exp \ (-1)... < a\} \rangle$ by (auto simp: continuous-on-def) also have at x within $\{-exp (-1).. < a\} = at$ -right x using a by (intro at-within-nhd[of - $\{..<a\}$]) (auto simp: True) also have $\ldots = at x$ within $\{-exp (-1) ... < 0\}$ using a by (intro at-within-nhd[of - $\{..<0\}$]) (auto simp: True) finally show ?thesis . qed qed qed **lemma** continuous-on-Lambert-W'-alt [continuous-intros]: assumes continuous-on A f $\bigwedge x. x \in A \implies f x \in \{-exp (-1)..<0\}$ **shows** continuous-on A (λx . Lambert-W' (f x)) using continuous-on-compose2 [OF continuous-on-Lambert-W' assms(1)] assms**by** (*auto simp: subset-iff*) **lemma** *tendsto-Lambert-W-1*: assumes $(f \longrightarrow L)$ F eventually $(\lambda x. f x \ge -exp(-1))$ F **shows** $((\lambda x. Lambert-W (f x)) \longrightarrow Lambert-W L) F$ **proof** (cases F = bot) **case** [*simp*]: *False* from tendsto-lowerbound [OF assms] have $L \ge -exp(-1)$ by simp thus *?thesis* using continuous-on-tendsto-compose[OF continuous-on-Lambert-W assms(1)] assms(2) by simpqed auto **lemma** tendsto-Lambert-W-2: assumes $(f \longrightarrow L) F L > -exp (-1)$ **shows** $((\lambda x. Lambert-W (f x)) \longrightarrow Lambert-W L) F$ using order-tendstoD(1)[OF assms] assms by (intro tendsto-Lambert-W-1) (auto elim: eventually-mono) **lemma** tendsto-Lambert-W [tendsto-intros]: assumes $(f \longrightarrow L)$ F eventually $(\lambda x. f x \ge -exp(-1))$ F $\lor L > -exp(-1)$ **shows** $((\lambda x. Lambert-W (f x)) \longrightarrow Lambert-W L) F$ using assms(2)proof assume L > -exp(-1)from order-tendstoD(1)[OF assms(1) this] assms(1) show ?thesis

by (*intro tendsto-Lambert-W-1*) (*auto elim: eventually-mono*) **qed** (*use tendsto-Lambert-W-1*[*OF assms*(1)] **in** *auto*)

lemma tendsto-Lambert-W'-1: assumes $(f \longrightarrow L)$ F eventually $(\lambda x. f x \ge -exp (-1))$ F L < 0shows $((\lambda x. Lambert-W'(f x)) \longrightarrow Lambert-W'L)$ Fproof (cases F = bot)case [simp]: False from tendsto-lowerbound[OF assms(1,2)] have L-ge: $L \ge -exp (-1)$ by simp from order-tendstoD(2)[OF assms(1,3)] have ev: eventually $(\lambda x. f x < 0)$ Fby autowith assms(2) have $eventually (\lambda x. f x \in \{-exp (-1)...<0\})$ Fby eventually-elim autothus ?thesis using L-ge assms(3)by $(intro \ continuous-on-tendsto-compose[OF \ continuous-on-Lambert-W' assms(1)])$ autoged auto

lemma tendsto-Lambert-W'-2:

assumes $(f \longrightarrow L) F L > -exp(-1) L < 0$ shows $((\lambda x. Lambert-W'(f x)) \longrightarrow Lambert-W'L) F$ using order-tendstoD(1)[OF assms(1,2)] assms by (intro tendsto-Lambert-W'-1) (auto elim: eventually-mono)

lemma tendsto-Lambert-W' [tendsto-intros]: **assumes** $(f \longrightarrow L)$ F eventually $(\lambda x. f x \ge -exp (-1))$ F $\lor L > -exp (-1)$ L < 0 **shows** $((\lambda x. Lambert-W' (f x)) \longrightarrow Lambert-W' L)$ F **using** assms(2) **proof assume** L > -exp (-1) **from** order-tendstoD(1)[OF assms(1) this] assms(1,3) **show** ?thesis **by** (intro tendsto-Lambert-W'-1) (auto elim: eventually-mono) **qed** (use tendsto-Lambert-W'-1[OF assms(1) - assms(3)] **in** auto)

lemma continuous-Lambert-W [continuous-intros]: **assumes** continuous F f f (Lim F (λx . x)) > -exp (-1) \lor eventually (λx . f x $\ge -exp (-1)$) F **shows** continuous F (λx . Lambert-W (f x)) **using** assms **unfolding** continuous-def **by** (intro tendsto-Lambert-W) auto

lemma continuous-Lambert-W' [continuous-intros]:

assumes continuous F f f (Lim $F (\lambda x. x)$) > $-exp(-1) \lor eventually (\lambda x. f x) \ge -exp(-1)) F$

 $f (Lim \ F \ (\lambda x. \ x)) < 0$

shows continuous $F(\lambda x. Lambert-W'(f x))$

using assms unfolding continuous-def by (intro tends to-Lambert-W') auto **lemma** has-field-derivative-Lambert-W [derivative-intros]: assumes x: x > -exp(-1)**shows** (Lambert-W has-real-derivative inverse (x + exp (Lambert-W x))) (at x within A) proof write Lambert-W $(\langle W \rangle)$ from x have W x > W (-exp (-1))**by** (subst Lambert-W-less-iff) auto hence W x > -1 by simp**note** [derivative-intros] = DERIV-inverse-function[where <math>g = Lambert-W] have $((\lambda x. x * exp x) has-real-derivative (1 + W x) * exp (W x)) (at (W x))$ **by** (*auto intro*!: *derivative-eq-intros simp*: *algebra-simps*) hence (W has-real-derivative inverse ((1 + W x) * exp (W x))) (at x) by (rule DERIV-inverse-function [where a = -exp(-1) and b = x + 1]) (use $x \langle W x \rangle -1$) in (auto simp: Lambert-W-times-exp-self Lim-ident-at *intro*!: *continuous-intros*) also have (1 + Wx) * exp(Wx) = x + exp(Wx)using x by (simp add: algebra-simps Lambert-W-times-exp-self) finally show ?thesis by (rule has-field-derivative-at-within) \mathbf{qed} **lemma** has-field-derivative-Lambert-W-gen [derivative-intros]: assumes (f has-real-derivative f') (at x within A) f x > -exp(-1)**shows** $((\lambda x. Lambert-W (f x)) has-real-derivative$ (f' / (f x + exp (Lambert-W (f x))))) (at x within A) using DERIV-chain2[OF has-field-derivative-Lambert-W[OF assms(2)] assms(1)] **by** (*simp add: field-simps*) **lemma** has-field-derivative-Lambert-W' [derivative-intros]: **assumes** *x*: $x \in \{-exp \ (-1) < .. < 0\}$ **shows** (Lambert-W' has-real-derivative inverse (x + exp (Lambert-W' x))) (at x within A) proof write Lambert- $W'(\langle W \rangle)$ from x have W x < W (-exp (-1))by (subst Lambert-W'-less-iff) auto hence W x < -1 by simp **note** [derivative-intros] = DERIV-inverse-function[**where**<math>g = Lambert-W] have $((\lambda x. x * exp x) has$ -real-derivative (1 + W x) * exp (W x)) (at (W x))**by** (*auto intro*!: *derivative-eq-intros simp*: *algebra-simps*) hence (W has-real-derivative inverse ((1 + W x) * exp (W x))) (at x)by (rule DERIV-inverse-function [where a = -exp(-1) and b = 0]) (use $x \langle W x \langle -1 \rangle$ in (auto simp: Lambert-W'-times-exp-self Lim-ident-at *intro*!: *continuous-intros*) also have (1 + Wx) * exp (Wx) = x + exp (Wx)using x by (simp add: algebra-simps Lambert-W'-times-exp-self)

finally show ?thesis by (rule has-field-derivative-at-within) qed

lemma has-field-derivative-Lambert-W'-gen [derivative-intros]:

assumes (f has-real-derivative f') (at x within A) $f x \in \{-exp \ (-1) < ... < 0\}$ shows ((λx . Lambert-W' (f x)) has-real-derivative

(f' / (f x + exp (Lambert-W'(f x))))) (at x within A) using DERU chain?[OF has field derivative Lambert W'[OF assert(2)].

using DERIV-chain2[OF has-field-derivative-Lambert-W'[OF assms(2)] assms(1)] **by** (simp add: field-simps)

1.6 Asymptotic expansion

Lastly, we prove some more detailed asymptotic expansions of W and W' at their singularities. First, we show that:

$$W(x) = \log x - \log \log x + o(\log \log x) \qquad \text{for } x \to \infty$$
$$W'(x) = \log(-x) - \log(-\log(-x)) + o(\log(-\log(-x))) \qquad \text{for } x \to 0^-$$

theorem Lambert-W-asymp-equiv-at-top: $(\lambda x. Lambert-W x - ln x) \sim [at-top] (\lambda x. -ln (ln x))$ proof – have $(\lambda x. Lambert-W x - \ln x) \sim [at-top] (\lambda x. (-1) * \ln (\ln x))$ **proof** (rule asymp-equiv-sandwich') fix c' :: real assume $c': c' \in \{-2 < .. < -1\}$ have eventually $(\lambda x. (\ln x + c' * \ln (\ln x)) * exp (\ln x + c' * \ln (\ln x)) \le x)$ at-top eventually $(\lambda x. \ln x + c' * \ln (\ln x) \ge -1)$ at-top using c' by real-asymp+ thus eventually (λx . Lambert- $W x - \ln x \ge c' * \ln (\ln x)$) at-top **proof** eventually-elim case (elim x) hence Lambert- $W x \ge \ln x + c' * \ln (\ln x)$ **by** (*intro* Lambert-W-geI) thus ?case by simp qed \mathbf{next} fix c' :: real assume c': $c' \in \{-1 < ... < 0\}$ have eventually $(\lambda x. (\ln x + c' * \ln (\ln x)) * exp (\ln x + c' * \ln (\ln x)) \ge x)$ at-top eventually $(\lambda x. \ln x + c' * \ln (\ln x) \ge -1)$ at-top using c' by real-asymp+ thus eventually (λx . Lambert- $W x - \ln x \leq c' * \ln (\ln x)$) at-top using eventually-ge-at-top[of -exp(-1)] proof eventually-elim case (elim x) hence Lambert-W $x \leq \ln x + c' * \ln (\ln x)$ **by** (*intro* Lambert-W-leI) thus ?case by simp qed

qed auto thus ?thesis by simp qed **lemma** Lambert-W-asymp-equiv-at-top' [asymp-equiv-intros]: Lambert- $W \sim [at-top] \ln$ proof – have $(\lambda x. Lambert-W x - \ln x) \in \Theta(\lambda x. -\ln (\ln x))$ **by** (*intro asymp-equiv-imp-bigtheta Lambert-W-asymp-equiv-at-top*) also have $(\lambda x::real. -ln (ln x)) \in o(ln)$ by real-asymp finally show ?thesis by (simp add: asymp-equiv-altdef) qed **theorem** Lambert-W'-asymp-equiv-at-left-0: $(\lambda x. Lambert-W' x - ln (-x)) \sim [at-left 0] (\lambda x. -ln (-ln (-x)))$ proof have $(\lambda x. Lambert-W' x - ln (-x)) \sim [at-left 0] (\lambda x. (-1) * ln (-ln (-x)))$ proof (rule asymp-equiv-sandwich') fix c':: real assume c': $c' \in \{-2 < .. < -1\}$ have eventually $(\lambda x. x \leq (\ln (-x) + c' * \ln (-\ln (-x))) * exp (\ln (-x) + c'))$ * ln (-ln (-x))) (at-left 0)eventually $(\lambda x::real. \ln (-x) + c' * \ln (-\ln (-x)) \leq -1)$ (at-left 0) eventually $(\lambda x::real. -exp(-1) \leq x)$ (at-left 0) using c' by real-asymp+ **thus** eventually $(\lambda x. Lambert-W'x - \ln(-x) \ge c' * \ln(-\ln(-x)))$ (at-left 0)**proof** eventually-elim case (elim x) hence Lambert-W' $x \ge ln (-x) + c' * ln (-ln (-x))$ by (intro Lambert-W'-geI) thus ?case by simp qed \mathbf{next} fix c' :: real assume $c': c' \in \{-1 < .. < 0\}$ have eventually $(\lambda x. x \ge (\ln (-x) + c' * \ln (-\ln (-x))) * exp (\ln (-x) + c'))$ * ln (-ln (-x))) (at-left 0)using c' by real-asymp moreover have eventually (λx ::real. x < 0) (at-left 0) **by** (*auto simp: eventually-at intro: exI[of - 1]*) ultimately show eventually (λx . Lambert-W' $x - \ln(-x) \leq c' * \ln(-\ln x)$ (-x))) (at-left θ) proof eventually-elim case (elim x) hence Lambert-W' $x \leq ln (-x) + c' * ln (-ln (-x))$ by (intro Lambert-W'-leI) thus ?case by simp ged qed auto thus ?thesis by simp

qed

lemma Lambert-W'-asymp-equiv'-at-left-0 [asymp-equiv-intros]: Lambert-W' ~[at-left 0] (λx . ln (-x)) **proof** – **have** (λx . Lambert-W' $x - \ln(-x)$) $\in \Theta[at-left 0](\lambda x. -ln (-ln (-x)))$ **by** (intro asymp-equiv-imp-bigtheta Lambert-W'-asymp-equiv-at-left-0) **also have** (λx ::real. -ln (-ln (-x))) $\in o[at-left 0](\lambda x. ln (-x))$ **by** real-asymp **finally show** ?thesis **by** (simp add: asymp-equiv-altdef) **qed**

Next, we look at the branching point $a := \frac{1}{e}$. Here, the asymptotic behaviour is as follows:

$$W(x) = -1 + \sqrt{2e}(x-a)^{\frac{1}{2}} - \frac{2}{3}e(x-a) + o(x-a) \qquad \text{for } x \to a^{+}$$
$$W'(x) = -1 - \sqrt{2e}(x-a)^{\frac{1}{2}} - \frac{2}{3}e(x-a) + o(x-a) \qquad \text{for } x \to a^{+}$$

lemma *sqrt-sqrt-mult*:

assumes $x \ge (0 :: real)$ shows $sqrt \ x * (sqrt \ x * y) = x * y$ using assms by (subst mult.assoc [symmetric]) auto

theorem Lambert-W-asymp-equiv-at-right-minus-exp-minus1:

defines $e \equiv exp \ 1$ defines $a \equiv -exp(-1)$ defines $C1 \equiv sqrt (2 * exp 1)$ defines $f \equiv (\lambda x. -1 + C1 * sqrt (x - a))$ shows $(\lambda x. Lambert-W x - f x) \sim [at-right a] (\lambda x. -2/3 * e * (x - a))$ proof define $C :: real \Rightarrow real$ where $C = (\lambda c. sqrt (2/e)/3 * (2*e+3*c))$ have asymp-equiv: $(\lambda x. (f x + c * (x - a)) * exp (f x + c * (x - a)) - x)$ \sim [at-right a] (λx . C c * (x - a) powr (3/2)) if $c \neq -2/3 * e$ for cproof – from that have $C c \neq 0$ by (auto simp: C-def e-def) have $(\lambda x. (f x + c * (x - a)) * exp (f x + c * (x - a)) - x - C c * (x - a))$ powr (3/2) $\in o[at\text{-right } a](\lambda x. (x - a) powr (3/2))$ unfolding f-def a-def C-def C1-def e-def by (real-asymp simp: field-simps real-sqrt-mult real-sqrt-divide sqrt-sqrt-mult exp-minus simp flip: sqrt-def) thus ?thesis using $\langle C \ c \neq 0 \rangle$ by (intro smallo-imp-asymp-equiv) auto qed show ?thesis

proof (rule asymp-equiv-sandwich')

fix c' :: real assume $c': c' \in \{-e < .. < -2/3 * e\}$ hence neq: $c' \neq -2/3 * e$ by auto from c' have neg: C c' < 0 unfolding C-def by (auto introl: mult-pos-neg) hence eventually (λx . C c' * (x - a) powr (3 / 2) < 0) (at-right a) **by** real-asymp hence eventually $(\lambda x. (f x + c' * (x - a)) * exp (f x + c' * (x - a)) - x < (x - a))$ 0) (at-right a) using asymp-equiv-eventually-neg-iff[OF asymp-equiv[OF neq]] by eventually-elim (use neg in auto) thus eventually $(\lambda x. Lambert-W x - f x \ge c' * (x - a))$ (at-right a) proof eventually-elim case (elim x) hence Lambert-W $x \ge f x + c' * (x - a)$ **by** (*intro* Lambert-W-geI) auto thus ?case by simp qed \mathbf{next} fix c' :: real assume $c': c' \in \{-2/3 * e < ... < 0\}$ hence neq: $c' \neq -2/3 * e$ by auto from c' have pos: C c' > 0 unfolding C-def by auto hence eventually $(\lambda x. C c' * (x - a) powr (3 / 2) > 0)$ (at-right a) by real-asymp hence eventually $(\lambda x. (f x + c' * (x - a)) * exp (f x + c' * (x - a)) - x >$ 0) (at-right a) using asymp-equiv-eventually-pos-iff[OF asymp-equiv[OF neq]] by eventually-elim (use pos in auto) **moreover have** eventually $(\lambda x. - 1 \le f x + c' * (x - a))$ (at-right a) eventually $(\lambda x. x > a)$ (at-right a) **unfolding** a-def f-def C1-def c' by real-asymp+ ultimately show eventually (λx . Lambert- $W x - f x \leq c' * (x - a)$) (at-right a)**proof** eventually-elim case (elim x) hence Lambert-W $x \leq f x + c' * (x - a)$ **by** (*intro Lambert-W-leI*) (*auto simp: a-def*) thus ?case by simp qed **qed** (auto simp: e-def) qed **theorem** Lambert-W'-asymp-equiv-at-right-minus-exp-minus1: defines $e \equiv exp \ 1$ defines $a \equiv -exp(-1)$ defines $C1 \equiv sqrt (2 * exp 1)$ defines $f \equiv (\lambda x. -1 - C1 * sqrt (x - a))$ shows $(\lambda x. Lambert-W' x - f x) \sim [at-right a] (\lambda x. -2/3 * e * (x - a))$ proof -

define $C :: real \Rightarrow real$ where $C = (\lambda c. -sqrt (2/e)/3 * (2*e+3*c))$

have asymp-equiv: $(\lambda x. (f x + c * (x - a)) * exp (f x + c * (x - a)) - x)$ \sim [at-right a] (λx . C c * (x - a) powr (3/2)) if $c \neq -2/3 * e$ for cproof – from that have $C c \neq 0$ **by** (*auto simp*: *C*-*def e*-*def*) have $(\lambda x. (f x + c * (x - a)) * exp (f x + c * (x - a)) - x - C c * (x - a))$ powr (3/2) $\in o[at\text{-right } a](\lambda x. (x - a) powr (3/2))$ unfolding f-def a-def C-def C1-def e-def by (real-asymp simp: field-simps real-sqrt-mult real-sqrt-divide sqrt-sqrt-mult exp-minus simp flip: sqrt-def) thus ?thesis using $\langle C \ c \neq 0 \rangle$ by (intro smallo-imp-asymp-equiv) auto qed show ?thesis **proof** (*rule asymp-equiv-sandwich'*) fix c' :: real assume $c': c' \in \{-e < .. < -2/3 * e\}$ hence neq: $c' \neq -2/3 * e$ by auto from c' have pos: C c' > 0 unfolding C-def by (auto introl: mult-pos-neg) hence eventually (λx . C c' * (x - a) powr (3 / 2) > 0) (at-right a) by real-asymp hence eventually $(\lambda x. (f x + c' * (x - a)) * exp (f x + c' * (x - a)) - x >$ 0) (at-right a) using asymp-equiv-eventually-pos-iff[OF asymp-equiv[OF neq]] by eventually-elim (use pos in auto) moreover have eventually $(\lambda x. x > a)$ (at-right a) eventually $(\lambda x. f x + c' * (x - a) \leq -1)$ (at-right a) **unfolding** *a-def f-def C1-def c'* **by** *real-asymp+* ultimately show eventually (λx . Lambert-W' $x - f x \ge c' * (x - a)$) (at-right a)**proof** eventually-elim case (elim x) hence Lambert-W' $x \ge f x + c' * (x - a)$ by (intro Lambert-W'-geI) (auto simp: a-def) thus ?case by simp qed next fix c' :: real assume $c': c' \in \{-2/3 * e < .. < 0\}$ hence neq: $c' \neq -2/3 * e$ by auto from c' have neg: C c' < 0 unfolding C-def by auto hence eventually (λx . C c' * (x - a) powr (3 / 2) < 0) (at-right a) by real-asymp hence eventually $(\lambda x. (f x + c' * (x - a)) * exp (f x + c' * (x - a)) - x < (x - a))$ 0 (at-right a) using asymp-equiv-eventually-neg-iff[OF asymp-equiv[OF neq]] by eventually-elim (use neg in auto) moreover have eventually ($\lambda x. x < 0$) (at-right a)

```
unfolding a-def by real-asymp

ultimately show eventually (\lambda x. Lambert-W' x - f x \le c' * (x - a)) (at-right

a)

proof eventually-elim

case (elim x)

hence Lambert-W' x \le f x + c' * (x - a)

by (intro Lambert-W'-leI) auto

thus ?case by simp

qed

qed

qed

(auto simp: e-def)

qed
```

Lastly, just for fun, we derive a slightly more accurate expansion of $W_0(x)$ for $x \to \infty$:

theorem Lambert-W-asymp-equiv-at-top": $(\lambda x. Lambert-W x - \ln x + \ln (\ln x)) \sim [at-top] (\lambda x. \ln (\ln x) / \ln x)$ proof have $(\lambda x. Lambert-W x - \ln x + \ln (\ln x)) \sim [at-top] (\lambda x. 1 * (\ln (\ln x) / \ln x))$ proof (rule asymp-equiv-sandwich') fix c' :: real assume $c': c' \in \{0 < .. < 1\}$ define a where $a = (\lambda x :: real. \ln x - \ln (\ln x) + c' * (\ln (\ln x) / \ln x))$ have eventually $(\lambda x. \ a \ x * exp \ (a \ x) \le x)$ at-top using c' unfolding *a-def* by *real-asymp+* thus eventually (λx . Lambert- $W x - \ln x + \ln (\ln x) \ge c' * (\ln (\ln x) / \ln x))$) at-top proof eventually-elim case (elim x) hence Lambert-W $x \ge a x$ **by** (*intro* Lambert-W-geI) thus ?case by (simp add: a-def) qed \mathbf{next} fix c' :: real assume $c': c' \in \{1 < .. < 2\}$ define a where $a = (\lambda x :: real. \ln x - \ln (\ln x) + c' * (\ln (\ln x) / \ln x))$ have eventually $(\lambda x. \ a \ x * exp \ (a \ x) \ge x)$ at-top eventually (λx . $a \ x \ge -1$) at-top using c' unfolding *a-def* by *real-asymp+* thus eventually $(\lambda x. \text{ Lambert-} W x - \ln x + \ln (\ln x) \le c' * (\ln (\ln x) / \ln x))$ at-top using eventually-ge-at-top[of -exp(-1)] **proof** eventually-elim case (elim x) hence Lambert-W x < a x**by** (*intro* Lambert-W-leI) thus ?case by (simp add: a-def) qed qed auto thus ?thesis by simp qed

 \mathbf{end}

```
theory Lambert-W-MacLaurin-Series
imports
HOL-Computational-Algebra.Formal-Power-Series
Bernoulli.Bernoulli-FPS
Stirling-Formula.Stirling-Formula
Lambert-W
begin
```

1.7 The MacLaurin series of $W_0(x)$ at x = 0

In this section, we derive the MacLaurin series of $W_0(x)$ as a formal power series at x = 0 and prove that its radius of convergence is e^{-1} .

We do not actually show that this series evaluates to 1 since Isabelle's library does not contain the required theorems about convergence of the composition of two power series yet. If it did, however, this last remaining step would be trivial since we did all the real work here.

lemma Stirling-Suc-n-n: Stirling (Suc n) n = (Suc n choose 2)by (induction n) (auto simp: choose-two)

The following defines the power series W(X) as the formal inverse of the formal power series Xe^X :

definition fps-Lambert-W :: real fps where fps-Lambert-W = fps-inv (fps-X * fps-exp 1)

The formal composition of W(X) and Xe^X is, in fact, the identity (in both directions).

lemma fps-compose-Lambert-W: fps-compose fps-Lambert-W (fps-X * fps-exp 1) = fps-X

unfolding fps-Lambert-W-def by (rule fps-inv) auto

lemma fps-compose-Lambert-W': fps-compose (fps-X * fps-exp 1) fps-Lambert-W = fps-X

unfolding fps-Lambert-W-def by (rule fps-inv-right) auto

We have W(0) = 0, which shows that W(X) indeed represents the branch W_0 .

lemma fps-nth-Lambert-W-0 [simp]: fps-nth fps-Lambert-W 0 = 0by (simp add: fps-Lambert-W-def fps-inv-def)

lemma fps-nth-Lambert-W-1 [simp]: fps-nth fps-Lambert-W 1 = 1by (simp add: fps-Lambert-W-def fps-inv-def)

lemma Stirling-n-n-minus-1: $n > 0 \implies$ Stirling n (n - 1) = (n choose 2)using Stirling-Suc-n-n[of n - 1] by (cases n) auto

All the equalities that hold for the analytic Lambert W function in a neighbourhood of 0 also hold formally for the formal power series, e.g. $W(X) = Xe^{-W(X)}$:

lemma *fps-Lambert-W-over-X*: fps-Lambert-W = fps-X * fps-compose (fps-exp(-1)) fps-Lambert-Wproof have fps-nth (fps-exp 1 oo fps-Lambert-W) 0 = 1by simp hence nz: fps-exp 1 oo fps-Lambert- $W \neq 0$ by force have fps-Lambert-W * fps-compose (fps-exp 1) fps-Lambert-W =fps-compose (fps-X * fps-exp 1) fps-Lambert-W**by** (*simp add: fps-compose-mult-distrib*) also have $\ldots = fps \cdot X * fps \cdot compose \ 1 \ fps \cdot Lambert \cdot W$ by (simp add: fps-compose-Lambert-W') also have 1 = fps-exp(-1) * fps-exp(1 :: real)**by** (*simp flip: fps-exp-add-mult*) also have fps-X * fps-compose ... fps-Lambert-W =fps-X * fps-compose (fps-exp (-1)) fps-Lambert-W *fps-compose (fps-exp 1) fps-Lambert-W **by** (*simp add: fps-compose-mult-distrib mult-ac*) finally show ?thesis using nz by simpqed

We now derive the closed-form expression

$$W(X) = \sum_{n=1}^{\infty} \frac{(-n)^{n-1}}{n!} X^n$$

lemma fps-nth-Lambert-W: fps-nth fps-Lambert-W n = (if n = 0 then 0 else((-n) (n-1) / fact n))proof define F :: real fps where F = fps-X * fps-exp 1 have fps-nth-eq: fps-nth F n = 1 / fact (n - 1) if n > 0 for n using that unfolding *F*-def by simp have F-power: $F \cap n = fps-X \cap n * fps-exp$ (of-nat n) for n by (simp add: F-def power-mult-distrib fps-exp-power-mult) have fps-nth (fps-inv F) $n = (if n = 0 then \ 0 else ((-n) (n-1) / fact n))$ for n **proof** (*induction n rule*: *less-induct*) case (less n) consider $n = 0 \mid n = 1 \mid n > 1$ by force thus ?case proof cases case 3hence fps-nth (fps-inv F) $n = -(\sum i=0..n-1.$ fps-nth (fps-inv F) i * fps-nth $(F \cap i) n$

(is - -?S) by (cases n) (auto simp: fps-inv-def F-def) also have $?S = (\sum i=1..< n. fps-nth (fps-inv F) i * fps-nth (F^i) n)$ using less[of 1] 3 by (intro sum.mono-neutral-right) (auto simp: not-le) also have $\dots = (-1)^{(n+1)} / fact n * (\sum_{i=1}^{n} (-1)^{(n-1)} (n-i) * real (n choose i) * real i^{(n-1)} (n-i) * real (n choose i) * real i^{(n-1)} (n-i) *$ 1))) unfolding sum-divide-distrib sum-distrib-left **proof** (*intro sum.cong*, *goal-cases*) case (2 i)hence fps-nth (fps-inv F) i * fps-nth ($F \uparrow i$) n = $(-1) \hat{(i-1)} * real (i \hat{(i-1)} * i \hat{(n-i)}) *$ (fact n / (fact i * fact (n - i)) / fact n)using less.IH[of i] by (simp add: F-power less fps-X-power-mult-nth power-minus') also have (fact n / (fact i * fact (n - i))) = real (n choose i)using 2 by (subst binomial-fact) auto also have i (i - 1) * i (n - i) = i (n - 1)using 2 by (subst power-add [symmetric]) auto also have $(-1) \hat{(i-1)} = ((-1) \hat{(n+1)} * (-1) \hat{(n-i)} :: real)$ using 2 by (subst power-add [symmetric]) (auto simp: minus-one-power-iff) finally show ?case by simp qed auto also have $(\sum i=1..< n. ((-1)(n-i) * real (n choose i) * real i (n-i)))$ (1))) = $(\sum i \in \{..n\} - \{n\}, ((-1))(n-i) * real (n choose i) * real i (n-i))$ 1))) using 3 by (intro sum.mono-neutral-left) auto also have $\ldots = (\sum i \le n. ((-1))(n-i) * real (n choose i) * real i (n-i))$ 1))) real $n \cap (n-1)$ by (subst (2) sum.remove[of - n]) auto also have $(\sum i \le n. ((-1)(n-i) * real (n choose i) * real i (n-1))) =$ real (Stirling (n - 1) n) * fact n by (subst Stirling-closed-form) auto also have Stirling (n - 1) n = 0using 3 by (subst Stirling-less) auto finally have fps-nth (fps-inv F) $n = -((-1) \cap n * real n \cap (n-1) / fact n)$ by simp also have $\ldots = (-real \ n) \ \widehat{} (n-1) \ / \ fact \ n$ using 3 by (subst power-minus) (auto simp: minus-one-power-iff) finally show ?thesis using 3 by simp **qed** (auto simp: fps-inv-def F-def) qed thus ?thesis by (simp add: F-def fps-Lambert-W-def) qed

Next, we need a few auxiliary lemmas about summability and convergence radii that should go into Isabelle's standard library at some point: **lemma** *summable-comparison-test-bigo*: fixes $f :: nat \Rightarrow real$ assumes summable $(\lambda n. norm (g n)) f \in O(g)$ **shows** summable f proof – **from** $\langle f \in O(g) \rangle$ **obtain** C where C: eventually $(\lambda x. norm (f x) \leq C * norm)$ (g x)) at-top by (auto elim: landau-o.bigE) thus ?thesis by (rule summable-comparison-test-ev) (insert assms, auto intro: summable-mult) qed lemma summable-comparison-test-bigo': assumes summable $(\lambda n. norm (g n))$ assumes $(\lambda n. norm (f n :: 'a :: banach)) \in O(\lambda n. norm (g n))$ **shows** summable f **proof** (rule summable-norm-cancel, rule summable-comparison-test-bigo) **show** summable $(\lambda n. norm (norm (g n)))$ using assms by simp $\mathbf{qed} \ fact+$ lemma conv-radius-conv-Sup': fixes $f :: nat \Rightarrow 'a :: \{banach, real-normed-div-algebra\}$ shows conv-radius $f = Sup \{r. \forall z. ereal (norm z) < r \longrightarrow summable (\lambda n. norm$ $(f n * z \cap n))$ **proof** (rule Sup-eqI [symmetric], goal-cases) case (1 r)show ?case **proof** (rule conv-radius-geI-ex') fix r' :: real assume r': r' > 0 ereal r' < rshow summable $(\lambda n. f n * of real r' \cap n)$ by (rule summable-norm-cancel) (use 1 r' in auto) \mathbf{qed} \mathbf{next} case (2 r)from 2[of 0] have r: r > 0 by *auto* show ?case **proof** (*intro conv-radius-leI-ex'* r) fix R assume R: R > 0 ereal R > rwith r obtain r' where [simp]: r = ereal r' by (cases r) auto **show** \neg summable ($\lambda n. f n * of-real R \cap n$) proof **assume** *: summable (λn . $f n * of-real R \cap n$) define R' where R' = (R + r') / 2from R have R': R' > r' R' < R by (simp-all add: R'-def) hence $\forall z. norm \ z < R' \longrightarrow summable \ (\lambda n. norm \ (f \ n \ast z \ n))$ using powser-insidea[OF *] by auto from 2[of R'] and this have $R' \leq r'$ by auto with $\langle R' > r' \rangle$ show False by simp

```
qed
  qed
qed
lemma bigo-imp-conv-radius-ge:
  fixes f g :: nat \Rightarrow 'a :: \{ banach, real-normed-field \}
 assumes f \in O(q)
  shows conv-radius f \ge conv-radius g
proof –
  have conv-radius g = Sup \{r. \forall z. ereal (norm z) < r \longrightarrow summable (\lambda n. norm
(g n * z \widehat{n})
   by (simp add: conv-radius-conv-Sup')
  also have \ldots \leq Sup \{r. \forall z. ereal (norm z) < r \longrightarrow summable (\lambda n. f n * z ^
n)\}
  proof (rule Sup-subset-mono, safe)
   fix r :: ereal and z :: 'a
   assume g: \forall z. ereal (norm z) < r \longrightarrow summable (\lambda n. norm (g n * z ^n))
   assume z: ereal (norm z) < r
   from g z have summable (\lambda n. norm (g n * z \cap n))
     by blast
   moreover have (\lambda n. norm (f n * z \cap n)) \in O(\lambda n. norm (g n * z \cap n))
     unfolding landau-o.big.norm-iff by (intro landau-o.big.mult assms) auto
   ultimately show summable (\lambda n. f n * z \cap n)
     by (rule summable-comparison-test-bigo')
  \mathbf{qed}
  also have \ldots = conv-radius f
   by (simp add: conv-radius-conv-Sup)
  finally show ?thesis .
qed
lemma conv-radius-cong-bigtheta:
  assumes f \in \Theta(g)
 shows conv-radius f = conv-radius g
 using assms
 by (intro antisym bigo-imp-conv-radius-ge) (auto simp: bigtheta-def bigomega-iff-bigo)
lemma conv-radius-eqI-smallomega-smallo:
  fixes f :: nat \Rightarrow 'a :: \{real-normed-div-algebra, banach\}
  assumes \Lambda \varepsilon. \varepsilon > l \Longrightarrow \varepsilon < inverse \ C \Longrightarrow (\lambda n. \ norm \ (f \ n)) \in \omega(\lambda n. \ \varepsilon \ \hat{n})
  assumes \Lambda \varepsilon. \varepsilon < u \Longrightarrow \varepsilon > inverse \ C \Longrightarrow (\lambda n. \ norm \ (f \ n)) \in o(\lambda n. \ \varepsilon \ \hat{n})
  assumes C: C > 0 and lu: l > 0 l < inverse C u > inverse C
  shows conv-radius f = ereal C
proof (intro antisym)
  have \theta < inverse C
   using assms by (auto simp: field-simps)
  also have \ldots < u
   by fact
```

finally have u > 0 by simp show conv-radius $f \ge C$

```
34
```

unfolding conv-radius-altdef le-Liminf-iff **proof** safe fix c :: ereal assume c: c < Chence $max \ c \ (inverse \ u) < ereal \ C$ using $lu \ C \langle u > 0 \rangle$ by (auto simp: field-simps) from ereal-dense2 [OF this] obtain c' where c': c < ereal c' inverse u < c' c'< Cby *auto* have inverse u > 0using $\langle u > 0 \rangle$ by simp also have $\ldots < c'$ by fact finally have c' > 0. have $\forall_F x \text{ in sequentially. norm } (norm (f x)) \leq 1/2 * norm (inverse c' \cap x)$ using landau-o.smallD[OF assms(2)][of inverse c'], of 1/2] c' C lu $\langle c' > 0 \rangle$ c **by** (*simp add: field-simps*) **thus** \forall_F *n* in sequentially. c < inverse (ereal (root n (norm (f n))))using eventually-gt-at-top[of 0] **proof** eventually-elim case (*elim* n) have norm $(f n) \leq 1/2 * norm$ (inverse $c' \cap n$) using c' using elim by (simp add: field-simps) also have ... < norm (inverse $c' \cap n$) using $\langle c' > \theta \rangle$ by simp finally have root n (norm (f n)) < root n (norm (inverse $c' \cap n$)) using $\langle n > 0 \rangle$ c' by (intro real-root-less-mono) auto also have root n (norm (inverse $c' \cap n$)) = inverse c'using $\langle n > 0 \rangle \langle c' > 0 \rangle$ by (simp add: norm-power real-root-power) finally have ereal (root n (norm (f n))) < ereal (inverse c') by simp also have \ldots = inverse (ereal c') using $\langle c' > 0 \rangle$ by *auto* finally have inverse (inverse (ereal c')) < inverse (ereal (root n (norm (f *n*)))) using $c' \langle n > 0 \rangle$ by (intro ereal-inverse-antimono-strict) auto also have inverse (inverse (ereal c')) = ereal c'using c' by simp finally show ?case using $\langle c < c' \rangle$ by simp qed qed \mathbf{next} show conv-radius $f \leq C$ **proof** (*rule ccontr*) assume $\neg(conv\text{-}radius f \leq C)$ hence conv-radius f > C by auto hence min (conv-radius f) (inverse l) > ereal C using $lu \ C \ \langle l > 0 \rangle$ by (auto simp: field-simps) from ereal-dense2[OF this] obtain c where c: C < ereal c inverse l > c c <

```
conv-radius f
     by auto
   hence c > 0 using lu C
     by (simp add: field-simps)
   have \forall_F n \text{ in sequentially. ereal } c < \text{inverse (ereal (root n (norm (f n))))}
     using less-LiminfD[OF c(3)[unfolded conv-radius-altdef]] by simp
   moreover have \forall_F n in sequentially. norm (f n) \geq 2 * norm (inverse c \cap n)
     using landau-omega.smallD[OF assms(1)[of inverse c], of 2] c C \langle c > 0 \rangle lu
     by (simp add: field-simps)
   ultimately have eventually (\lambda n. False) sequentially
     using eventually-gt-at-top[of 0]
   proof eventually-elim
     case (elim \ n)
     have norm (inverse c \cap n) < 2 * norm (inverse c \cap n)
       using c \langle n > 0 \rangle C by simp
     also have \ldots \leq norm (f n)
      using elim by simp
     finally have root n (inverse c \cap n) < root n (norm (f n))
       using \langle n > 0 \rangle by (intro real-root-less-mono) auto
     also have root n (inverse c \cap n) = inverse c
       using \langle n > 0 \rangle c C by (subst real-root-power) auto
     finally have ereal (inverse c) < ereal (root n (norm (f n)))
      by simp
     also have ereal (inverse c) = inverse (ereal c)
      using c \ C by auto
     finally have inverse (ereal (root n (norm (f n)))) < inverse (inverse (ereal
c))
      using c C
      by (intro ereal-inverse-antimono-strict) auto
     also have \ldots = ereal c
      using c \ C by auto
     also have \ldots < inverse (ereal (root n (norm (f n))))
      using elim by simp
     finally show False.
   qed
   thus False by simp
  qed
qed
```

Finally, we show that the radius of convergence of W(X) is e^{-1} by directly computing

$$\lim_{n \to \infty} \sqrt[n]{|[X^n] W(X)|} = e$$

using Stirling's formula for n!:

lemma fps-conv-radius-Lambert-W: fps-conv-radius fps-Lambert-W = exp(-1) **proof** –

have conv-radius (fps-nth fps-Lambert-W) = conv-radius ($\lambda n. exp \ 1 \ n * n \text{ powr} (-3/2) :: real$)

proof (*rule conv-radius-conq-bigtheta*)

have fps-nth fps-Lambert- $W \in \Theta(\lambda n. (-real n) \cap (n-1) / fact n)$ by (intro bigthetaI-cong eventually-mono[OF eventually-gt-at-top[of 0]]) (*auto simp: fps-nth-Lambert-W*)

also have $(\lambda n. (-real n) \cap (n-1) / fact n) \in \Theta(\lambda n. real n \cap (n-1) / fact$ n)

by (subst landau-theta.norm-iff [symmetric], subst norm-divide) auto

also have $(\lambda n. (real n) \uparrow (n-1) / fact n) \in \Theta(\lambda n. (real n) \uparrow (n-1) / (sqrt (2 * pi * real n) * (real n / exp 1))$ (n)

by (*intro asymp-equiv-imp-bigtheta asymp-equiv-intros fact-asymp-equiv*)

also have $(\lambda n. (real n) \cap (n-1) / (sqrt (2 * pi * real n) * (real n / exp 1))$ $(n) \in n$

 $\Theta(\lambda n. exp \ 1 \ \widehat{} n * n \ powr \ (-3/2))$

by (*real-asymp simp*: *ln-inverse*)

finally show fps-nth fps-Lambert- $W \in \Theta(\lambda n. exp \ 1 \ n * n \ powr \ (-3/2) ::$ real).

qed

also have ... = inverse (limsup (λn . ereal (root n (exp 1 ^ n * real n powr -(3 / 2)))))

by (simp add: conv-radius-def)

also have limsup $(\lambda n. ereal (root n (exp 1 \cap n * real n powr - (3 / 2)))) = exp$ 1

proof (*intro lim-imp-Limsup tendsto-intros*)

- real asymp does not support *root* for a variable basis natively, so we need to convert it to (powr) first.

have $(\lambda n. (exp \ 1 \cap n * real \ n \ powr \ -(3/2))$ powr $(1 / real \ n)) \longrightarrow exp \ 1$ **by** real-asymp

also have ?this $\leftrightarrow (\lambda x. root \ x \ (exp \ 1 \ x * real \ x \ powr \ - \ (3 \ / \ 2))) \longrightarrow$ exp 1

by (intro filterlim-cong eventually-mono[OF eventually-gt-at-top[of 0]]) (auto simp: root-powr-inverse)

finally show

qed auto

finally show ?thesis

by (*simp add: fps-conv-radius-def exp-minus*) qed

end

References

[1] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. Advances in Computational Math*ematics*, 5(1):329–359, Dec. 1996.