

# The Transcendence of $e$

Manuel Eberl

May 26, 2024

## Abstract

This work contains a formalisation of the proof that Euler's number  $e$  is transcendental. The proof follows the standard approach of assuming that  $e$  is algebraic and then using a specific integer polynomial to derive two inconsistent bounds, leading to a contradiction.

This approach can be found in many different sources; this formalisation mostly follows a PlanetMath article [1] by Roger Lipsett.

## Contents

<b>1</b>	<b>Proof of the Transcendence of <math>e</math></b>	<b>1</b>
1.1	Various auxiliary facts . . . . .	1
1.2	General facts about polynomials . . . . .	2
1.3	Main proof . . . . .	3

## 1 Proof of the Transcendence of $e$

**theory** *E-Transcendental*

**imports**

*HOL-Complex-Analysis.Complex-Analysis*

*HOL-Number-Theory.Number-Theory*

*HOL-Computational-Algebra.Polynomial*

*Polynomial-Interpolation.Ring-Hom-Poly*

**begin**

**hide-const (open)** *UnivPoly.coeff UnivPoly.up-ring.monom*

**hide-const (open)** *Module.smult Coset.order*

### 1.1 Various auxiliary facts

**lemma** *fact-dvd-pochhammer*:

**assumes**  $m \leq n + 1$

**shows** *fact m dvd pochhammer (int n - int m + 1) m*

**proof** –

**have** *(real n gchoose m) \* fact m = of-int (pochhammer (int n - int m + 1) m)*

**by** *(simp add: gbinomial-pochhammer' pochhammer-of-int [symmetric])*

**also have**  $(\text{real } n \text{ gchoose } m) * \text{fact } m = \text{of-int } (\text{int } (n \text{ choose } m) * \text{fact } m)$   
**by**  $(\text{simp add: binomial-gbinomial})$   
**finally have**  $\text{int } (n \text{ choose } m) * \text{fact } m = \text{pochhammer } (\text{int } n - \text{int } m + 1) m$   
**by**  $(\text{subst } (\text{asm}) \text{ of-int-eq-iff})$   
**from this [symmetric] show ?thesis by simp**  
**qed**

**lemma prime-elem-int-not-dvd-neg1-power:**  
 $\text{prime-elem } (p :: \text{int}) \implies \neg p \text{ dvd } (-1) ^ n$   
**by**  $(\text{metis dvdI minus-one-mult-self unit-imp-no-prime-divisors})$

**lemma nat-fact [simp]:**  $\text{nat } (\text{fact } n) = \text{fact } n$   
**by**  $(\text{metis nat-int of-nat-fact of-nat-fact})$

**lemma prime-dvd-fact-iff-int:**  
 $p \text{ dvd fact } n \iff p \leq \text{int } n \text{ if prime } p$   
**using**  $\text{that prime-dvd-fact-iff [of nat |p| n]}$   
**by auto (simp add: prime-ge-0-int)**

**lemma power-over-fact-tendsto-0:**  
 $(\lambda n. (x :: \text{real}) ^ n / \text{fact } n) \longrightarrow 0$   
**using**  $\text{summable-exp[of x] by (intro summable-LIMSEQ-zero) (simp add: sums-iff field-simps)}$

**lemma power-over-fact-tendsto-0':**  
 $(\lambda n. c * (x :: \text{real}) ^ n / \text{fact } n) \longrightarrow 0$   
**using**  $\text{tendsto-mult[OF tendsto-const[of c] power-over-fact-tendsto-0[of x]] by simp}$

## 1.2 General facts about polynomials

**lemma fact-dvd-higher-pderiv:**  
 $[:\text{fact } n :: \text{int}:] \text{dvd } (\text{pderiv } ^ n) p$   
**proof** –  
**have**  $[:\text{fact } n:] \text{dvd } (\text{pderiv } ^ n) (\text{monom } c k)$  **for**  $c :: \text{int}$  **and**  $k :: \text{nat}$   
**by**  $(\text{cases } n \leq k + 1)$   
 $(\text{simp-all add: higher-pderiv-monom higher-pderiv-monom-eq-zero fact-dvd-pochhammer const-poly-dvd-iff})$   
**hence**  $[:\text{fact } n:] \text{dvd } (\text{pderiv } ^ n) (\sum_{k \leq \text{degree } p} \text{monom } (\text{coeff } p k) k)$   
**by**  $(\text{simp-all add: higher-pderiv-sum dvd-sum})$   
**thus ?thesis by (simp add: poly-as-sum-of-monom)**  
**qed**

**lemma fact-dvd-poly-higher-pderiv-aux:**  
 $(\text{fact } n :: \text{int}) \text{dvd poly } ((\text{pderiv } ^ n) p) x$   
**proof** –  
**have**  $[:\text{fact } n:] \text{dvd } (\text{pderiv } ^ n) p$  **by**  $(\text{rule fact-dvd-higher-pderiv})$   
**then obtain q where**  $(\text{pderiv } ^ n) p = [:\text{fact } n:] * q$  **by**  $(\text{erule dvdE})$   
**thus ?thesis by simp**

qed

**lemma** *fact-dvd-poly-higher-pderiv-aux'*:  
 $m \leq n \implies (\text{fact } m :: \text{int}) \text{ dvd poly } ((\text{pderiv } \overset{\sim}{\sim} n) p) x$   
**by** (*meson dvd-trans fact-dvd fact-dvd-poly-higher-pderiv-aux*)

### 1.3 Main proof

**lemma** *lindemann-weierstrass-integral*:  
**fixes**  $u :: \text{complex}$  **and**  $f :: \text{complex poly}$   
**defines**  $df \equiv \lambda n. (\text{pderiv } \overset{\sim}{\sim} n) f$   
**defines**  $m \equiv \text{degree } f$   
**defines**  $I \equiv \lambda f u. \text{exp } u * (\sum_{j \leq \text{degree } f} \text{poly } ((\text{pderiv } \overset{\sim}{\sim} j) f) 0) -$   
 $(\sum_{j \leq \text{degree } f} \text{poly } ((\text{pderiv } \overset{\sim}{\sim} j) f) u)$   
**shows**  $((\lambda t. \text{exp } (u - t) * \text{poly } f t) \text{ has-contour-integral } I f u) (\text{linepath } 0 u)$   
**proof** –  
**note** [*derivative-intros*] =  
*exp-scaleR-has-vector-derivative-right vector-diff-chain-within*  
**let**  $?g = \lambda t. 1 - t$  **and**  $?f = \lambda t. -\text{exp } (t *_R u)$   
**have**  $((\lambda t. \text{exp } ((1 - t) *_R u) * u) \text{ has-integral } (\text{?f } \circ \text{?g}) 1 - (\text{?f } \circ \text{?g}) 0) \{0..1\}$   
**by** (*rule fundamental-theorem-of-calculus*)  
(*auto intro!*: *derivative-eq-intros simp del: o-apply*)  
**hence** *aux-integral*:  $((\lambda t. \text{exp } (u - t *_R u) * u) \text{ has-integral } \text{exp } u - 1) \{0..1\}$   
**by** (*simp add: algebra-simps*)  
  
**have**  $((\lambda t. \text{exp } (u - t *_R u) * u * \text{poly } f (t *_R u)) \text{ has-integral } I f u) \{0..1\}$   
**unfolding** *df-def m-def*  
**proof** (*induction degree f arbitrary: f*)  
**case** 0  
**then obtain**  $c$  **where**  $c: f = [c]$  **by** (*auto elim: degree-eq-zeroE*)  
**have**  $((\lambda t. c * (\text{exp } (u - t *_R u) * u)) \text{ has-integral } c * (\text{exp } u - 1)) \{0..1\}$   
**using** *aux-integral* **by** (*rule has-integral-mult-right*)  
**with**  $c$  **show** *?case* **by** (*simp add: algebra-simps I-def*)  
**next**  
**case** (*Suc m*)  
**define**  $df$  **where**  $df = (\lambda j. (\text{pderiv } \overset{\sim}{\sim} j) f)$   
**show** *?case*  
**proof** (*rule integration-by-parts[OF bounded-bilinear-mult]*)  
**fix**  $t :: \text{real}$  **assume**  $t \in \{0..1\}$   
**have**  $((\text{?f } \circ \text{?g}) \text{ has-vector-derivative } \text{exp } (u - t *_R u) * u) (\text{at } t)$   
**by** (*auto intro!*: *derivative-eq-intros simp: algebra-simps simp del: o-apply*)  
**thus**  $((\lambda t. -\text{exp } (u - t *_R u)) \text{ has-vector-derivative } \text{exp } (u - t *_R u) * u)$   
(*at t*)  
**by** (*simp add: algebra-simps o-def*)  
**next**  
**fix**  $t :: \text{real}$  **assume**  $t \in \{0..1\}$   
**have**  $(\text{poly } f \circ (\lambda t. t *_R u) \text{ has-vector-derivative } u * \text{poly } (\text{pderiv } f) (t *_R u))$   
(*at t*)

**by** (*rule field-vector-diff-chain-at*) (*auto intro!*: *derivative-eq-intros*)  
**thus**  $((\lambda t. \text{poly } f (t *_{\mathbb{R}} u)) \text{ has-vector-derivative } u * \text{poly } (pderiv f) (t *_{\mathbb{R}} u))$   
(at  $t$ )  
**by** (*simp add*: *o-def*)  
**next**  
**from** *Suc*(2) **have**  $m: m = \text{degree } (pderiv f)$  **by** (*simp add*: *degree-pderiv*)  
**from** *Suc*(1)[*OF this*] **this**  
**have**  $((\lambda t. \text{exp } (u - t *_{\mathbb{R}} u) * u * \text{poly } (pderiv f) (t *_{\mathbb{R}} u)) \text{ has-integral } \text{exp } u * (\sum_{j=0..m} \text{poly } (df (Suc j)) 0) - (\sum_{j=0..m} \text{poly } (df (Suc j)) u)) \{0..1\})$   
**by** (*simp add*: *df-def funpow-swap1 atMost-atLeast0 I-def*)  
**also have**  $(\sum_{j=0..m} \text{poly } (df (Suc j)) 0) = (\sum_{j=Suc 0..Suc m} \text{poly } (df j) 0)$   
**by** (*rule sum.shift-bounds-cl-Suc-ivl [symmetric]*)  
**also have**  $\dots = (\sum_{j=0..Suc m} \text{poly } (df j) 0) - \text{poly } f 0$   
**by** (*subst* (2) *sum.atLeast-Suc-atMost*) (*simp-all add*: *df-def*)  
**also have**  $(\sum_{j=0..m} \text{poly } (df (Suc j)) u) = (\sum_{j=Suc 0..Suc m} \text{poly } (df j) u)$   
**by** (*rule sum.shift-bounds-cl-Suc-ivl [symmetric]*)  
**also have**  $\dots = (\sum_{j=0..Suc m} \text{poly } (df j) u) - \text{poly } f u$   
**by** (*subst* (2) *sum.atLeast-Suc-atMost*) (*simp-all add*: *df-def*)  
**finally have**  $((\lambda t. - (\text{exp } (u - t *_{\mathbb{R}} u) * u * \text{poly } (pderiv f) (t *_{\mathbb{R}} u))) \text{ has-integral } -(\text{exp } u * ((\sum_{j=0..Suc m} \text{poly } (df j) 0) - \text{poly } f 0) - ((\sum_{j=0..Suc m} \text{poly } (df j) u) - \text{poly } f u))) \{0..1\})$   
**(is** (*- has-integral ?I*) **)** **by** (*rule has-integral-neg*)  
**also have**  $?I = - \text{exp } (u - 1 *_{\mathbb{R}} u) * \text{poly } f (1 *_{\mathbb{R}} u) - \text{exp } (u - 0 *_{\mathbb{R}} u) * \text{poly } f (0 *_{\mathbb{R}} u) - I f u$   
**by** (*simp add*: *df-def algebra-simps Suc*(2) *atMost-atLeast0 I-def*)  
**finally show**  $((\lambda t. - \text{exp } (u - t *_{\mathbb{R}} u) * (u * \text{poly } (pderiv f) (t *_{\mathbb{R}} u))) \text{ has-integral } \dots) \{0..1\}$  **by** (*simp add*: *algebra-simps*)  
**qed** (*auto intro!*: *continuous-intros*)  
**qed**  
**thus** *?thesis* **by** (*simp add*: *has-contour-integral-linepath algebra-simps*)  
**qed**

**locale** *lindemann-weierstrass-aux* =

**fixes**  $f :: \text{complex poly}$

**begin**

**definition**  $I :: \text{complex} \Rightarrow \text{complex}$  **where**

$$I u = \text{exp } u * (\sum_{j \leq \text{degree } f} \text{poly } ((pderiv \overset{\sim}{j} f) 0) - (\sum_{j \leq \text{degree } f} \text{poly } ((pderiv \overset{\sim}{j} f) u))$$

**lemma** *lindemann-weierstrass-integral-bound*:

**fixes**  $u :: \text{complex}$

**assumes**  $C \geq 0 \wedge t. t \in \text{closed-segment } 0 u \implies \text{norm } (\text{poly } f t) \leq C$

**shows**  $\text{norm } (I u) \leq \text{norm } u * \text{exp } (\text{norm } u) * C$

**proof** –

```

have I u = contour-integral (linepath 0 u) (λt. exp (u - t) * poly f t)
  using contour-integral-unique[OF lindemann-weierstrass-integral[of u f]] un-
folding I-def ..
also have norm ... ≤ exp (norm u) * C * norm (u - 0)
proof (intro contour-integral-bound-linepath)
  fix t assume t: t ∈ closed-segment 0 u
  then obtain s where s: s ∈ {0..1} t = s *R u by (auto simp: closed-segment-def)
  hence s * norm u ≤ 1 * norm u by (intro mult-right-mono) simp-all
  with s have norm-t: norm t ≤ norm u by auto

from s have Re u - Re t = (1 - s) * Re u by (simp add: algebra-simps)
also have ... ≤ norm u
proof (cases Re u ≥ 0)
  case True
    with ⟨s ∈ {0..1}⟩ have (1 - s) * Re u ≤ 1 * Re u by (intro mult-right-mono)
  simp-all
    also have Re u ≤ norm u by (rule complex-Re-le-cmod)
    finally show ?thesis by simp
  next
    case False
      with ⟨s ∈ {0..1}⟩ have (1 - s) * Re u ≤ 0 by (intro mult-nonneg-nonpos)
  simp-all
    also have ... ≤ norm u by simp
    finally show ?thesis .
qed
finally have exp (Re u - Re t) ≤ exp (norm u) by simp

hence exp (Re u - Re t) * norm (poly f t) ≤ exp (norm u) * C
  using assms t norm-t by (intro mult-mono) simp-all
thus norm (exp (u - t) * poly f t) ≤ exp (norm u) * C
  by (simp add: norm-mult exp-diff norm-divide field-simps)
qed (auto simp: intro!: mult-nonneg-nonneg contour-integrable-continuous-linepath
  continuous-intros assms)
finally show ?thesis by (simp add: mult-ac)
qed

end

```

```

lemma poly-higher-pderiv-aux1:
  fixes c :: 'a :: idom
  assumes k < n
  shows poly ((pderiv  $\overset{\sim}{\sim}$  k) ([:-c, 1:]  $\overset{\wedge}{\wedge}$  n * p)) c = 0
  using assms
proof (induction k arbitrary: n p)
  case (Suc k n p)
  from Suc.prem1 obtain n' where n: n = Suc n' by (cases n) auto
  from Suc.prem2 n have k < n' by simp
  have (pderiv  $\overset{\sim}{\sim}$  Suc k) ([:- c, 1:]  $\overset{\wedge}{\wedge}$  n * p) =
    (pderiv  $\overset{\sim}{\sim}$  k) ([:- c, 1:]  $\overset{\wedge}{\wedge}$  n * pderiv p + [:- c, 1:]  $\overset{\wedge}{\wedge}$  n' * smult (of-nat

```

$n$ )  $p$ )  
 by (simp only: funpow-Suc-right o-def pderiv-mult n pderiv-power-Suc,  
 simp only: n [symmetric]) (simp add: pderiv-pCons mult-ac)  
 also from Suc.premis  $\langle k < n' \rangle$  have poly ...  $c = 0$   
 by (simp add: higher-pderiv-add Suc.IH del: mult-smult-right)  
 finally show ?case .  
 qed simp-all

**lemma poly-higher-pderiv-aux1':**  
 fixes  $c :: 'a :: idom$   
 assumes  $k < n$   $[: -c, 1:] \hat{=}^n dvd p$   
 shows poly  $((pderiv \hat{=}^k) p) c = 0$   
**proof** -  
 from assms(2) obtain  $q$  where  $p = [: -c, 1:] \hat{=}^n * q$  by (elim dvdE)  
 also from assms(1) have poly  $((pderiv \hat{=}^k) \dots) c = 0$   
 by (rule poly-higher-pderiv-aux1)  
 finally show ?thesis .  
 qed

**lemma poly-higher-pderiv-aux2:**  
 fixes  $c :: 'a :: \{idom, semiring-char-0\}$   
 shows poly  $((pderiv \hat{=}^n) ([: -c, 1:] \hat{=}^n * p)) c = fact n * poly p c$   
**proof** (induction n arbitrary: p)  
 case (Suc n p)  
 have  $(pderiv \hat{=}^{Suc n} ([: -c, 1:] \hat{=}^{Suc n} * p)) =$   
 $(pderiv \hat{=}^n ([: -c, 1:] \hat{=}^{Suc n} * pderiv p)) +$   
 $(pderiv \hat{=}^n ([: -c, 1:] \hat{=}^n * smult (1 + of-nat n) p))$   
 by (simp del: funpow.simps power-Suc add: funpow-Suc-right pderiv-mult  
 pderiv-power-Suc higher-pderiv-add pderiv-pCons mult-ac)  
 also have  $[: -c, 1:] \hat{=}^{Suc n} * pderiv p = [: -c, 1:] \hat{=}^n * ([: -c, 1:] * pderiv p)$   
 by (simp add: algebra-simps)  
 finally show ?case by (simp add: Suc.IH del: mult-smult-right power-Suc)  
 qed simp-all

**lemma poly-higher-pderiv-aux3:**  
 fixes  $c :: 'a :: \{idom, semiring-char-0\}$   
 assumes  $k \geq n$   
 shows  $\exists q. poly ((pderiv \hat{=}^k) ([: -c, 1:] \hat{=}^n * p)) c = fact n * poly q c$   
 using assms  
**proof** (induction k arbitrary: n p)  
 case (Suc k n p)  
 show ?case  
**proof** (cases n)  
 fix  $n'$  assume  $n: n = Suc n'$   
 have poly  $((pderiv \hat{=}^{Suc k} ([: -c, 1:] \hat{=}^n * p)) c =$   
 $poly ((pderiv \hat{=}^k) ([: -c, 1:] \hat{=}^n * pderiv p)) c +$   
 $of-nat n * poly ((pderiv \hat{=}^k) ([: -c, 1:] \hat{=}^{n'} * p)) c$   
 by (simp del: funpow.simps power-Suc add: funpow-Suc-right pderiv-power-Suc  
 pderiv-mult n pderiv-pCons higher-pderiv-add mult-ac higher-pderiv-smult)

**also have**  $\exists q1. \text{poly } ((\text{pderiv } \widehat{\sim} k) ([: -c, 1:] \widehat{\sim} n * \text{pderiv } p)) c = \text{fact } n * \text{poly } q1 c$   
**using** *Suc.prem*s *Suc.IH*[of *n pderiv p*]  
**by** (*cases*  $n' = k$ ) (*auto simp: n poly-higher-pderiv-aux1 simp del: power-Suc of-nat-Suc*)  
*intro: exI*[of - 0::'a *poly*])  
**then obtain** *q1*  
**where**  $\text{poly } ((\text{pderiv } \widehat{\sim} k) ([: -c, 1:] \widehat{\sim} n * \text{pderiv } p)) c = \text{fact } n * \text{poly } q1 c ..$   
**also from** *Suc.IH*[of  $n' p$ ] *Suc.prem*s **obtain** *q2*  
**where**  $\text{poly } ((\text{pderiv } \widehat{\sim} k) ([: -c, 1:] \widehat{\sim} n' * p)) c = \text{fact } n' * \text{poly } q2 c$   
**by** (*auto simp: n*)  
**finally show** *?case* **by** (*auto intro!: exI*[of - *q1 + q2*] *simp: n algebra-simps*)  
**qed** *auto*  
**qed** *auto*

**lemma** *poly-higher-pderiv-aux3'*:  
**fixes**  $c :: 'a :: \{idom, semiring-char-0\}$   
**assumes**  $k \geq n$   $[: -c, 1:] \widehat{\sim} n \text{ dvd } p$   
**shows**  $\text{fact } n \text{ dvd } \text{poly } ((\text{pderiv } \widehat{\sim} k) p) c$   
**proof** -  
**from** *assms*(2) **obtain** *q* **where**  $p = [: -c, 1:] \widehat{\sim} n * q$  **by** (*elim dvdE*)  
**with** *poly-higher-pderiv-aux3*[*OF* *assms*(1), of *c q*] **show** *?thesis* **by** *auto*  
**qed**

**lemma** *e-transcendental-aux-bound*:  
**obtains** *C* **where**  $C \geq 0$   
 $\bigwedge x. x \in \text{closed-segment } 0 \text{ (of-nat } n) \implies$   
 $\text{norm } (\prod_{k \in \{1..n\}} (x - \text{of-nat } k :: \text{complex})) \leq C$   
**proof** -  
**let**  $?f = \lambda x. (\prod_{k \in \{1..n\}} (x - \text{of-nat } k))$   
**define** *C* **where**  $C = \max 0 (\text{Sup } (cmod \text{ ' } ?f \text{ ' closed-segment } 0 \text{ (of-nat } n)))$   
**have**  $C \geq 0$  **by** (*simp add: C-def*)  
**moreover** {  
**fix**  $x :: \text{complex}$  **assume**  $x \in \text{closed-segment } 0 \text{ (of-nat } n)$   
**hence**  $cmod (?f x) \leq \text{Sup } ((cmod \circ ?f) \text{ ' closed-segment } 0 \text{ (of-nat } n))$   
**by** (*intro cSup-upper bounded-imp-bdd-above compact-imp-bounded compact-continuous-image*)  
*(auto intro!: continuous-intros)*  
**also have**  $\dots \leq C$  **by** (*simp add: C-def image-comp*)  
**finally have**  $cmod (?f x) \leq C$ .  
**}**  
**ultimately show** *?thesis* **by** (*rule that*)  
**qed**

**theorem** *e-transcendental-complex*:  $\neg \text{algebraic } (exp 1 :: \text{complex})$   
**proof**  
**assume** *algebraic* (*exp 1 :: complex*)  
**then obtain**  $q :: \text{int poly}$   
**where**  $q: q \neq 0 \text{ coeff } q 0 \neq 0 \text{ poly } (of-int-poly q) (exp 1 :: \text{complex}) = 0$

by (elim algebraicE'-nonzero) simp-all

**define**  $n :: \text{nat}$  **where**  $n = \text{degree } q$

**from**  $q$  **have** [simp]:  $n \neq 0$  **by** (intro notI) (auto simp: n-def elim!: degree-eq-zeroE)

**define**  $qmax$  **where**  $qmax = \text{Max } (\text{insert } 0 \ (\text{abs } ' \text{set } (\text{coeffs } q)))$

**have**  $qmax\text{-nonneg}$  [simp]:  $qmax \geq 0$  **by** (simp add: qmax-def)

**have**  $qmax$ :  $|\text{coeff } q \ k| \leq qmax$  **for**  $k$

by (cases  $k \leq \text{degree } q$ )

(auto simp: qmax-def coeff-eq-0 coeffs-def simp del: upt-Suc intro: Max.coboundedI)

**obtain**  $C$  **where**  $C: C \geq 0$

$\bigwedge x. x \in \text{closed-segment } 0 \ (\text{of-nat } n) \implies \text{norm } (\prod_{k \in \{1..n\}}. (x - \text{of-nat } k :: \text{complex})) \leq C$

by (erule e-transcendental-aux-bound)

**define**  $E$  **where**  $E = (1 + \text{real } n) * \text{real-of-int } qmax * \text{real } n * \text{exp } (\text{real } n) / \text{real } n$

**define**  $F$  **where**  $F = \text{real } n * C$

**have**  $\text{ineq}$ :  $\text{fact } (p - 1) \leq E * F \wedge p$  **if**  $p$ :  $\text{prime } p \ p > n \ p > \text{abs } (\text{coeff } q \ 0)$  **for**  $p$

**proof** –

**from**  $p(1)$  **have**  $p\text{-pos}$ :  $p > 0$  **by** (simp add: prime-gt-0-nat)

**define**  $f :: \text{int poly}$

**where**  $f = \text{monom } 1 \ (p - 1) * (\prod_{k \in \{1..n\}}. [:-\text{of-nat } k, 1:] \wedge p)$

**have**  $\text{poly-f}$ :  $\text{poly } (\text{of-int-poly } f) \ x = x \wedge (p - 1) * (\prod_{k \in \{1..n\}}. (x - \text{of-nat } k)) \wedge p$

**for**  $x :: \text{complex}$  **by** (simp add: f-def poly-prod poly-monom prod-power-distrib hom-distrib)

**define**  $m :: \text{nat}$  **where**  $m = \text{degree } f$

**from**  $p\text{-pos}$  **have**  $m$ :  $m = (n + 1) * p - 1$

**by** (simp add: m-def f-def degree-mult-eq degree-monom-eq degree-prod-sum-eq degree-linear-power)

**define**  $M :: \text{int}$  **where**  $M = (-1) \wedge (n * p) * \text{fact } n \wedge p$

**with**  $p$  **have**  $p\text{-not-dvd-M}$ :  $\neg \text{int } p \ \text{dvd } M$

**by** (auto simp: M-def prime-elem-int-not-dvd-neg1-power prime-dvd-power-iff prime-gt-0-nat prime-dvd-fact-iff-int prime-dvd-mult-iff)

**have** [simp]:  $\text{poly } (\text{of-int-poly } p) \ (\text{complex-of-nat } k) = \text{of-int } (\text{poly } p \ (\text{int } k))$  **for**  $p \ k$

by (metis of-int-hom.poly-map-poly of-int-of-nat-eq)

**interpret**  $\text{lindemann-weierstrass-aux } \text{of-int-poly } f$  .

**define**  $J :: \text{complex}$  **where**  $J = (\sum_{k \leq n}. \text{of-int } (\text{coeff } q \ k) * I \ (\text{of-nat } k))$

**define**  $\text{idxs}$  **where**  $\text{idxs} = (\{..n\} \times \{..m\}) - \{(0, p - 1)\}$

**hence**  $J = (\sum_{k \leq n}. \text{of-int } (\text{coeff } q \ k) * \text{exp } 1 \wedge k) * (\sum_{n \leq m}. \text{of-int } (\text{poly } ((pderiv \wedge n) f) \ 0)) - \text{of-int } (\sum_{k \leq n}. \sum_{n \leq m}. \text{coeff } q \ k * \text{poly } ((pderiv \wedge n) f) \ (\text{int } k))$

**by** (simp add: J-def I-def algebra-simps sum-subtractf sum-distrib-left m-def



*exp-of-nat-mult [symmetric] flip: of-int-hom.map-poly-higher-pderiv*

**also have**  $(\sum k \leq n. \text{of-int } (\text{coeff } q \ k) * \text{exp } 1 \ ^k) = \text{poly } (\text{of-int-poly } q) (\text{exp } 1$   
 $:: \text{complex})$

**by** (*simp add: poly-altdef n-def*)

**also have**  $\dots = 0$  **by fact**

**finally have**  $J = \text{of-int } (-\sum (k,n) \in \{..n\} \times \{..m\}. \text{coeff } q \ k * \text{poly } ((\text{pderiv } \sim n) f) (\text{int } k))$

**by** (*simp add: sum.cartesian-product*)

**also have**  $\{..n\} \times \{..m\} = \text{insert } (0, p - 1) \ \text{idxs}$  **by** (*auto simp: m idxs-def*)

**also have**  $-\sum (k,n) \in \dots \text{coeff } q \ k * \text{poly } ((\text{pderiv } \sim n) f) (\text{int } k) =$   
 $-(\text{coeff } q \ 0 * \text{poly } ((\text{pderiv } \sim (p - 1)) f) \ 0) -$   
 $(\sum (k, n) \in \text{idxs}. \text{coeff } q \ k * \text{poly } ((\text{pderiv } \sim n) f) (\text{of-nat } k))$

**by** (*subst sum.insert*) (*simp-all add: idxs-def*)

**also have**  $\text{coeff } q \ 0 * \text{poly } ((\text{pderiv } \sim (p - 1)) f) \ 0 = \text{coeff } q \ 0 * M * \text{fact } (p - 1)$

**proof** –

**have**  $f = [:-0, 1:] \ ^{(p - 1)} * (\prod k = 1..n. [:- \text{of-nat } k, 1:] \ ^p)$

**by** (*simp add: f-def monom-altdef*)

**also have**  $\text{poly } ((\text{pderiv } \sim (p - 1)) \dots) \ 0 =$   
 $\text{fact } (p - 1) * \text{poly } (\prod k = 1..n. [:- \text{of-nat } k, 1:] \ ^p) \ 0$

**by** (*rule poly-higher-pderiv-aux2*)

**also have**  $\text{poly } (\prod k = 1..n. [:- \text{of-nat } k :: \text{int}, 1:] \ ^p) \ 0 = (-1) \ ^{(n*p)} * \text{fact } n \ ^p$

**by** (*induction n*) (*simp-all add: prod.nat-ivl-Suc' power-mult-distrib mult-ac power-minus' power-add del: of-nat-Suc*)

**finally show** *?thesis* **by** (*simp add: mult-ac M-def*)

**qed**

**also obtain**  $N$  **where**  $(\sum (k, n) \in \text{idxs}. \text{coeff } q \ k * \text{poly } ((\text{pderiv } \sim n) f) (\text{int } k)) = \text{fact } p * N$

**proof** –

**have**  $\forall (k, n) \in \text{idxs}. \text{fact } p \ \text{dvd} \ \text{poly } ((\text{pderiv } \sim n) f) (\text{of-nat } k)$

**proof clarify**

**fix**  $k \ j$  **assume**  $\text{idxs}: (k, j) \in \text{idxs}$

**then consider**  $k = 0 \ j < p - 1 \mid k = 0 \ j > p - 1 \mid k \neq 0 \ j < p \mid k \neq 0 \ j \geq p$

**by** (*fastforce simp: idxs-def*)

**thus**  $\text{fact } p \ \text{dvd} \ \text{poly } ((\text{pderiv } \sim j) f) (\text{of-nat } k)$

**proof cases**

**case 1**

**thus** *?thesis*

**by** (*simp add: f-def poly-higher-pderiv-aux1' monom-altdef*)

**next**

**case 2**

**thus** *?thesis*

**by** (*simp add: f-def poly-higher-pderiv-aux3' monom-altdef fact-dvd-poly-higher-pderiv-aux'*)

**next**

**case 3**

**thus** *?thesis unfolding f-def*

**by** (*subst poly-higher-pderiv-aux1 '[of - p]*)

```

      (insert idxs, auto simp: idxs-def intro!: dvd-mult)
    next
      case 4
      thus ?thesis unfolding f-def
        by (intro poly-higher-pderiv-aux3') (insert idxs, auto intro!: dvd-mult
simp: idxs-def)
      qed
      qed
      hence fact p dvd ( $\sum (k, n) \in \text{idxs}. \text{coeff } q \ k * \text{poly } ((\text{pderiv } \sim n) f) (\text{int } k)$ )
        by (auto intro!: dvd-sum dvd-mult simp del: of-int-fact)
      with that show thesis
        by blast
      qed
      also from p have  $-(\text{coeff } q \ 0 * M * \text{fact } (p - 1)) - \text{fact } p * N =$ 
         $-\text{fact } (p - 1) * (\text{coeff } q \ 0 * M + p * N)$ 
        by (subst fact-reduce[of p]) (simp-all add: algebra-simps)
      finally have  $J: J = -\text{of-int } (\text{fact } (p - 1) * (\text{coeff } q \ 0 * M + p * N))$  by simp

      from p q(2) have  $\neg p \text{ dvd } \text{coeff } q \ 0 * M + p * N$ 
      by (auto simp: dvd-add-left-iff p-not-dvd-M prime-dvd-fact-iff-int prime-dvd-mult-iff
dest: dvd-imp-le-int)
      hence  $\text{coeff } q \ 0 * M + p * N \neq 0$  by (intro notI) simp-all
      hence  $\text{abs } (\text{coeff } q \ 0 * M + p * N) \geq 1$  by simp
      hence  $\text{norm } (\text{of-int } (\text{coeff } q \ 0 * M + p * N) :: \text{complex}) \geq 1$  by (simp only:
norm-of-int)
      hence  $\text{fact } (p - 1) * \dots \geq \text{fact } (p - 1) * 1$  by (intro mult-left-mono) simp-all
      hence  $J\text{-lower: } \text{norm } J \geq \text{fact } (p - 1)$  unfolding J norm-minus-cancel
of-int-mult of-int-fact
      by (simp add: norm-mult)

      have  $\text{norm } J \leq (\sum k \leq n. \text{norm } (\text{of-int } (\text{coeff } q \ k) * I (\text{of-nat } k)))$ 
      unfolding J-def by (rule norm-sum)
      also have  $\dots \leq (\sum k \leq n. \text{of-int } q_{\max} * (\text{real } n * \exp (\text{real } n) * \text{real } n ^ (p -$ 
1) *  $C ^ p))$ 
      proof (intro sum-mono)
        fix k assume k:  $k \in \{..n\}$ 
        have  $n > 0$  by (rule ccontr) simp
        {
          fix x :: complex assume x:  $x \in \text{closed-segment } 0 (\text{of-nat } k)$ 
          then obtain t where  $t: t \geq 0 \ t \leq 1 \ x = \text{of-real } t * \text{of-nat } k$ 
          by (auto simp: closed-segment-def scaleR-conv-of-real)
          hence  $\text{norm } x = t * \text{real } k$  by (simp add: norm-mult)
          also from  $\langle t \leq 1 \rangle k$  have  $*$ :  $\dots \leq 1 * \text{real } n$  by (intro mult-mono) simp-all
          finally have  $x'$ :  $\text{norm } x \leq \text{real } n$  by simp
          from  $t \langle n > 0 \rangle *$  have  $x''$ :  $x \in \text{closed-segment } 0 (\text{of-nat } n)$ 
          by (auto simp: closed-segment-def scaleR-conv-of-real field-simps
intro!: exI[of - t * real k / real n] )
          have  $\text{norm } (\text{poly } (\text{of-int-poly } f) x) =$ 
             $\text{norm } x ^ (p - 1) * \text{cmod } (\prod_{i=1..n} x - i) ^ p$ 

```

by (*simp add: poly-f norm-mult norm-power*)  
 also from  $x x' x''$  have  $\dots \leq \text{of-nat } n^{(p-1)} * C^p$   
 by (*intro mult-mono C power-mono simp-all*)  
 finally have  $\text{norm } (\text{poly } (\text{of-int-poly } f) x) \leq \text{real } n^{(p-1)} * C^p$ .  
 } note  $A = \text{this}$

have  $\text{norm } (I (\text{of-nat } k)) \leq$   
 $\text{cmod } (\text{of-nat } k) * \text{exp } (\text{cmod } (\text{of-nat } k)) * (\text{of-nat } n^{(p-1)} * C^p)$   
 by (*intro lindemann-weierstrass-integral-bound[OF - A]*  
*C mult-nonneg-nonneg zero-le-power*) *auto*  
 also have  $\dots \leq \text{cmod } (\text{of-nat } n) * \text{exp } (\text{cmod } (\text{of-nat } n)) * (\text{of-nat } n^{(p-1)} * C^p)$   
 using  $k$  by (*intro mult-mono zero-le-power mult-nonneg-nonneg C*) *simp-all*  
 finally show  $\text{cmod } (\text{of-int } (\text{coeff } q k) * I (\text{of-nat } k)) \leq$   
 $\text{of-int } q\text{max} * (\text{real } n * \text{exp } (\text{real } n) * \text{real } n^{(p-1)} * C^p)$   
 unfolding *norm-mult*  
 by (*intro mult-mono*) (*simp-all add: qmax of-int-abs [symmetric] del: of-int-abs*)  
 qed  
 also have  $\dots = E * F^p$  using *p-pos*  
 by (*simp add: power-diff power-mult-distrib E-def F-def*)  
 finally show  $\text{fact } (p-1) \leq E * F^p$  using *J-lower* by *linarith*  
 qed

have  $(\lambda n. E * F * F^{(n-1)} / \text{fact } (n-1)) \longrightarrow 0$  (*is ?P*)  
 by (*intro filterlim-compose[OF power-over-fact-tendsto-0' filterlim-minus-const-nat-at-top]*)  
 also have  $?P \longleftrightarrow (\lambda n. E * F^n / \text{fact } (n-1)) \longrightarrow 0$   
 by (*intro filterlim-cong refl eventually-mono[OF eventually-gt-at-top[of 0::nat]]*)  
*(auto simp: power-Suc [symmetric] simp del: power-Suc)*  
 finally have *eventually*  $(\lambda n. E * F^n / \text{fact } (n-1) < 1)$  *at-top*  
 by (*rule order-tendstoD*) *simp-all*  
 hence *eventually*  $(\lambda n. E * F^n < \text{fact } (n-1))$  *at-top* by *eventually-elim simp*  
 then obtain  $P$  where  $P: \bigwedge n. n \geq P \implies E * F^n < \text{fact } (n-1)$   
 by (*auto simp: eventually-at-top-linorder*)

have  $\exists p. \text{prime } p \wedge p > \text{Max } \{\text{nat } (\text{abs } (\text{coeff } q 0)), n, P\}$  by (*rule bigger-prime*)  
 then obtain  $p$  where  $\text{prime } p \wedge p > \text{Max } \{\text{nat } (\text{abs } (\text{coeff } q 0)), n, P\}$  by *blast*  
 hence  $\text{int } p > \text{abs } (\text{coeff } q 0) \wedge p > n \wedge p \geq P$  by *auto*  
 with *ineq[of p] <prime p>* have  $\text{fact } (p-1) \leq E * F^p$  by *simp*  
 moreover from  $\langle p \geq P \rangle$  have  $\text{fact } (p-1) > E * F^p$  by (*rule P*)  
 ultimately show *False* by *linarith*  
 qed

**corollary** *e-transcendental-real:  $\neg$  algebraic (exp 1 :: real)*  
**proof** –  
 have  $\neg \text{algebraic } (\text{exp } 1 :: \text{complex})$  by (*rule e-transcendental-complex*)  
 also have  $(\text{exp } 1 :: \text{complex}) = \text{of-real } (\text{exp } 1)$  using *exp-of-real[of 1]* by *simp*  
 also have *algebraic*  $\dots \longleftrightarrow \text{algebraic } (\text{exp } 1 :: \text{real})$  by *simp*

finally show *?thesis* .  
qed

end

## References

- [1] R. Lipsett. Planetmath. <http://planetmath.org/prooffindemannweierstrassentheoremundthateandpiaretranscendental>, 2007.