The Abel–Ruffini theorem and the unsolvability of the quintic

Lawrence C. Paulson

Abstract

This entry proves the *Abel–Ruffini theorem* — a polynomial equation solvable by radicals has a solvable Galois group — and applies it to a concrete quintic to conclude that the general quintic is not solvable by radicals. We follow the corresponding *mathlib* formalisation [2].

The forward direction — *solvable by radicals implies solvable Galois group* — is assembled on the session `New_Algebra`: cyclotomic and Kummer extensions have solvable (indeed abelian) Galois groups, solvability is preserved by the tower correspondence, and the normal closure of a radical tower is again a radical tower obtained by splicing the Galois conjugates of the original tower.

For the concrete witness we study the rational quintic $\Phi = x^5 - 4x + 2$ and show that the Galois group of its splitting field over $\mathbb{Q}$ is not solvable. We prove that Φ is irreducible over $\mathbb{Q}$ (Eisenstein's criterion at 2) and, using the AFP entry `Sturm_Sequences` [1], show that it has exactly three real roots and one conjugate pair of non-real roots. Complex conjugation therefore acts on the five roots as a transposition, while irreducibility makes the Galois action transitive; a transitive subgroup of S_5 containing a transposition is the whole of S_5, which is not solvable. Combining the two halves yields the headline result that Φ is not solvable by radicals.

Contents

1 Cyclotomic and Kummer extensions have abelian Galois groups 4

2 Roots of unity form a cyclic group 4

2.1 The set of nth roots of unity 4

2.2 The primitive root generates: powers exhaust the roots of unity . 4

2.3 Closure properties: a subgroup of the multiplicative group 5

2.4 Divisibility: fewer roots of unity sit among more 5

3 Cyclotomic and Kummer extensions have abelian Galois groups 6

3.1 Automorphisms map the carrier into itself 6

3.2 Abelian criterion for a simple normal extension 6

3.3 Cyclotomic extensions are abelian . . . 7
3.4 Kummer extensions are abelian . . . 9

4 The forward Abel–Ruffini direction: assembling the radical tower 10
4.1 A single radical step, with roots of unity present, is a full splitting field . . . 10
4.2 A Kummer step is both solvable and stable . . . 11
4.3 A cyclotomic step is solvable, stable, and supplies the roots of unity 12
4.4 A chain of Kummer steps is a solvable tower . . . 14
4.5 Solvability descends to a stable subfield with surjective restriction 16
4.6 The forward theorem . . . 17
4.7 From a roots-of-unity-free chain to a valid Kummer chain . . . 17
4.8 Extracting Kummer witnesses from a radical tower . . . 19
4.9 Extracting a nondegenerate witness list from a bare radical tower 19
4.10 A radical tower over a roots-of-unity base has a solvable Galois group . . . 21
4.11 Unconditional descent to a stable splitting subfield . . . 22

5 The normal radical closure of a radical tower 23
5.1 A Galois conjugate of a radical tower is a radical tower . . . 23
5.2 The family of all conjugates is a family of towers over the base . 23
5.3 The spliced tower's top contains every tower of the family . . . 24
5.4 The normal radical closure exists . . . 25
5.5 Every element of a radical tower top is algebraic over the base . 26
5.6 A splitting field is normal and has a finite Galois group . . . 26
5.7 Prepending a cyclotomic step . . . 27
5.8 A radical tower embeds in a solvable normal radical top . . . 27
5.9 Polynomial helpers: products over a subfield and their roots . . . 30
5.10 Witness-level normal closure: bounded-exponent Kummer chain for the conjugate family . . . 31
5.11 The normal radical closure of a radical tower is a solvable splitting field . . . 35

6 A Concrete Quintic with Galois Group S_5 41
6.1 Eisenstein's irreducibility criterion . . . 42
6.2 The polynomial and its real roots . . . 43
6.3 The real polynomial and its real roots . . . 44
6.4 The complex roots: three real and one conjugate pair . . . 44
6.5 Irreducibility of the quintic over the rationals . . . 47

6.6 The Galois group of the quintic is not solvable 47
6.7 The main theorem 48

7 The quintic $x^5 - 4x + 2$ is not solvable by radicals **50**
7.1 The main theorem 50

1 Cyclotomic and Kummer extensions have abelian Galois groups

theory *Cyclotomic-Kummer*
imports *HOL−Analysis.Complex-Transcendental*
HOL−New-Algebra.Subfield
HOL−New-Algebra.Galois-Degree
begin

2 Roots of unity form a cyclic group

For the complex numbers there is an *explicit* primitive nth root of unity, *unity-root* n = *exp* (*complex-of-real* (2 ∗ *pi*) ∗ i / n), and every nth root of unity is one of its powers. Thus the group of nth roots of unity is *cyclic*, generated by *unity-root n*.

2.1 The set of nth roots of unity

definition *roots-of-unity* :: *nat* ⇒ *complex set*
where *roots-of-unity n* ≡ {*z*. *z* ^ *n* = *1*}

A distinguished primitive nth root of unity.

definition *unity-root* :: *nat* ⇒ *complex*
where *unity-root n* ≡ *exp* (2 ∗ *of-real pi* ∗ i / *of-nat n*)

lemma *roots-of-unity-iff* [*simp*]: *z* ∈ *roots-of-unity n* ⟷ *z* ^ *n* = *1*
by (*simp add*: *roots-of-unity-def*)

lemma *one-in-roots-of-unity* [*simp*, *intro*]: *1* ∈ *roots-of-unity n*
by *simp*

lemma *finite-roots-of-unity*:
assumes *n* > *0* **shows** *finite* (*roots-of-unity n*)
unfolding *roots-of-unity-def* **using** *assms* **by** (*intro finite-roots-unity*) *simp*

lemma *card-roots-of-unity*:
assumes *n* > *0* **shows** *card* (*roots-of-unity n*) = *n*
unfolding *roots-of-unity-def* **using** *assms* **by** (*simp add*: *card-complex-roots-unity*)

2.2 The primitive root generates: powers exhaust the roots of unity

The jth power of *unity-root n* is *exp* (2πij / n), the standard parametrisation of the nth roots of unity.

lemma *unity-root-power*:
unity-root n ^ *j* = *exp* (2 ∗ *of-real pi* ∗ i ∗ *of-nat j* / *of-nat n*)

by (*metis exp-of-nat2-mult times-divide-eq-left unity-root-def*)

Each power of the primitive root is indeed an nth root of unity.

lemma *unity-root-pow-mem*:
assumes $n > 0$ **shows** *unity-root* $n \mathbin{\hat{}} j \in$ *roots-of-unity* n
by (*simp add*: *assms complex-root-unity unity-root-power*)

Cyclicity. The nth roots of unity are exactly the powers $\{(\textit{unity-root}\ n)^j \mid j.\ j < n\}$ of the primitive root: the group is cyclic.

theorem *roots-of-unity-eq-powers*:
assumes $n > 0$
shows *roots-of-unity* $n = \{\textit{unity-root}\ n \mathbin{\hat{}} j \mid j.\ j < n\}$
using *assms complex-roots-unity roots-of-unity-def unity-root-power* **by** *auto*

corollary *roots-of-unity-subset-powers*:
assumes $n > 0$ **and** $z \in$ *roots-of-unity* n
shows $\exists j < n.\ z =$ *unity-root* $n \mathbin{\hat{}} j$
using *assms roots-of-unity-eq-powers* **by** *blast*

2.3 Closure properties: a subgroup of the multiplicative group

lemma *unity-root-mem* [*simp*, *intro*]:
assumes $n > 0$ **shows** *unity-root* $n \in$ *roots-of-unity* n
using *unity-root-pow-mem*[*OF assms*, *of 1*] **by** *simp*

lemma *roots-of-unity-mult-closed*:
assumes $x \in$ *roots-of-unity* n **and** $y \in$ *roots-of-unity* n
shows $x * y \in$ *roots-of-unity* n
using *assms* **by** (*simp add*: *power-mult-distrib*)

lemma *roots-of-unity-nonzero*:
assumes $n > 0$ **and** $z \in$ *roots-of-unity* n **shows** $z \neq 0$
by (*metis assms roots-of-unity-iff zero-neq-one zero-power*)

lemma *roots-of-unity-inverse-closed*:
assumes $z \in$ *roots-of-unity* n **shows** *inverse* $z \in$ *roots-of-unity* n
by (*metis assms inverse-1 power-inverse roots-of-unity-iff*)

2.4 Divisibility: fewer roots of unity sit among more

If n divides N then every nth root of unity is an Nth root of unity: from $z \mathbin{\hat{}} n = 1$ and $N = n * k$ we get $z \mathbin{\hat{}} N = (z \mathbin{\hat{}} n) \mathbin{\hat{}} k = 1$. Hence one primitive Nth root of unity supplies the roots of unity for every exponent dividing N — the fact that lets a single cyclotomic step serve a whole radical tower.

lemma *roots-of-unity-dvd-mono*:
fixes z :: *complex*
assumes *dvd*: n *dvd* N **and** z: $z \mathbin{\hat{}} n = 1$

shows $z \mathbin{\hat{}} N = 1$
by (*metis dvd dvdE power-mult power-one z*)

corollary *roots-of-unity-subset-dvd*:
assumes *n dvd N*
shows *roots-of-unity n* $\subseteq$ *roots-of-unity N*
using *assms roots-of-unity-dvd-mono* **by** (*auto simp*: *roots-of-unity-def*)

3 Cyclotomic and Kummer extensions have abelian Galois groups

Both the cyclotomic extension $F(\zeta)$ (adjoining a primitive root of unity) and the Kummer extension $F(a)$ with $a^n \in F$ and the nth roots of unity already in F have *abelian* Galois groups, hence *solvable* ones. Both follow from one criterion: the Galois group of a simple normal extension is abelian as soon as its automorphisms commute on the generator, and an F-automorphism of $F\ a$ is determined by its value at a."

3.1 Automorphisms map the carrier into itself

An F-automorphism of *eval-img F a* keeps its value at a inside the carrier (it is extensional into *eval-img F a*).

lemma *field-auto-gen-closed*:
fixes *F* :: *complex set* **and** *a* :: *complex*
assumes *Subfield F* $\sigma \in$ *field-auto* (*eval-img F a*) *F*
shows $\sigma\ a \in$ *eval-img F a*
using *Subfield.eval-img-self field-auto-closed assms* **by** *blast*

3.2 Abelian criterion for a simple normal extension

If any two F-automorphisms of *eval-img F a* commute at the generator a, then *field-auto* (*eval-img F a*) *F* is an abelian group. Determinacy (*field-auto-determined-by-gen*) upgrades commutation at a to full commutation, since both *compose E* σ τ and *compose E* τ σ are automorphisms agreeing at a.

lemma *galois-abelian-of-gen-commute*:
fixes *F* :: *complex set* **and** *a* :: *complex*
assumes *sfF*: *Subfield F* **and** *alg*: *algebraic-over F a*
and *comm-gen*: $\bigwedge\sigma\ \tau.\ \sigma \in$ *field-auto* (*eval-img F a*) *F* $\Longrightarrow \tau \in$ *field-auto* (*eval-img F a*) *F*
$\Longrightarrow \sigma\ (\tau\ a) = \tau\ (\sigma\ a)$
shows *Abelian-Group* (*field-auto* (*eval-img F a*) *F*) (*compose* (*eval-img F a*)) (*identity* (*eval-img F a*))
proof −
let *?E* = *eval-img F a*
have *sfE*: *Subfield ?E* **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])

have *cE*: *complex-subfield ?E* **using** *sfE* **by** (*simp add*: *complex-subfield-iff-subfield*)
have *FE*: $F \subseteq$ *?E* **using** *Subfield.eval-img-base*[*OF sfF*] **by** *blast*
have *aE*: $a \in$ *?E* **by** (*rule Subfield.eval-img-self*[*OF sfF*])
interpret *G*: *Group field-auto ?E F compose ?E identity ?E*
by (*rule Galois-group-Group*[*OF cE FE*])
— Commutativity of the group, obtained from commutation at a via determinacy.
have *comm*: *compose ?E* σ τ = *compose ?E* τ σ
if *s*: $\sigma \in$ *field-auto ?E F* **and** *t*: $\tau \in$ *field-auto ?E F* **for** σ τ
proof −
obtain *st*: *compose ?E* σ $\tau \in$ *field-auto ?E F* **and** *ts*: *compose ?E* τ $\sigma \in$ *field-auto ?E F*
using *s t* **by** *blast*
— Both composites send a to the same value.
have *compose ?E* σ τ *a* = σ (τ *a*) **using** *aE* **by** (*simp add*: *compose-eq*)
also have ... = τ (σ *a*) **using** *comm-gen*[*OF s t*] **.**
also have ... = *compose ?E* τ σ *a* **using** *aE* **by** (*simp add*: *compose-eq*)
finally have *compose ?E* σ τ *a* = *compose ?E* τ σ *a* **.**
then show *?thesis*
by (*rule field-auto-determined-by-gen*[*OF sfF alg st ts*])
qed
have *cm*: *commutative-monoid* (*field-auto ?E F*) (*compose ?E*) (*identity ?E*)
using *G.Monoid-axioms comm*
by (*simp add*: *commutative-monoid-def commutative-monoid-axioms-def*)
show *?thesis*
by (*rule Abelian-Group.intro*[*OF G.Group-axioms cm*])
qed

Consequently such a Galois group is *solvable*.

theorem *galois-solvable-of-gen-commute*:
fixes *F* :: *complex set* **and** *a* :: *complex*
assumes *sfF*: *Subfield F* **and** *alg*: *algebraic-over F a*
and *comm-gen*: $\bigwedge\sigma\ \tau.\ \sigma \in$ *field-auto* (*eval-img F a*) *F* $\Longrightarrow$ $\tau \in$ *field-auto* (*eval-img F a*) *F*
$\Longrightarrow$ σ (τ *a*) = τ (σ *a*)
shows *Group.solvable* (*field-auto* (*eval-img F a*) *F*) (*compose* (*eval-img F a*)) (*identity* (*eval-img F a*))
proof −
interpret *AB*: *Abelian-Group field-auto* (*eval-img F a*) *F*
compose (*eval-img F a*) *identity* (*eval-img F a*)
by (*simp add*: *alg comm-gen galois-abelian-of-gen-commute sfF*)
show *?thesis*
using *AB.abelian-imp-solvable AB.commutative-monoid-axioms* **by** *blast*
qed

3.3 Cyclotomic extensions are abelian

The polynomial $X^n - 1$ over F, and the fact that an automorphism sends an nth root of unity to an nth root of unity.

lemma *field-auto-maps-unity-root*:

fixes F :: *complex set* **and** z :: *complex*
assumes *sfF*: *Subfield F* **and** *n*: $n > 0$ **and** *alg*: *algebraic-over F z*
and *s*: $\sigma \in$ *field-auto* (*eval-img F z*) *F* **and** *root*: $z \hat{} n = 1$
shows $(\sigma\ z) \hat{} n = 1$
proof −
let *?E* = *eval-img F z*
have *sfE*: *Subfield ?E* **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])
have *FE*: $F \subseteq$ *?E* **using** *Subfield.eval-img-base*[*OF sfF*] **by** *blast*
have *zE*: $z \in$ *?E* **by** (*rule Subfield.eval-img-self*[*OF sfF*])
define *P* **where** $P \equiv$ (*monom 1 n − 1* :: *complex poly*)
have *PF*: $P \in$ *poly-over F*
by (*simp add*: *P-def sfF Subfield.one-closed Subfield.poly-over-1 Subfield.poly-over-diff Subfield.poly-over-monom*)
have *poly P x* = $x \hat{} n - 1$ **for** *x* **by** (*simp add*: *P-def poly-monom*)
then show *?thesis*
using *field-auto-maps-root*[*OF sfE sfF FE s PF zE*] *root* **by** *simp*
qed

Cyclotomic case. If ζ is a *primitive* nth root of unity — every nth root of unity is a power of it — then $F\ \zeta$ has an abelian Galois group: each automorphism sends ζ to a power ζ^k, and powers commute.

theorem *cyclotomic-galois-solvable*:
fixes F :: *complex set* **and** ζ :: *complex*
assumes *sfF*: *Subfield F* **and** *n*: $n > 0$ **and** *alg*: *algebraic-over F* ζ
and *root*: $\zeta \hat{} n = 1$
and *primitive*: $\bigwedge w.\ w \hat{} n = 1 \Longrightarrow \exists k.\ w = \zeta \hat{} k$
shows *Group.solvable* (*field-auto* (*eval-img F* ζ) *F*) (*compose* (*eval-img F* ζ)) (*identity* (*eval-img F* ζ))
proof (*rule galois-solvable-of-gen-commute*[*OF sfF alg*])
fix $\sigma\ \tau$ **assume** *s*: $\sigma \in$ *field-auto* (*eval-img F* ζ) *F* **and** *t*: $\tau \in$ *field-auto* (*eval-img F* ζ) *F*
have *sfE*: *Subfield* (*eval-img F* ζ) **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])
then obtain *homs*: *field-hom-on* (*eval-img F* ζ) σ **and** *homt*: *field-hom-on* (*eval-img F* ζ) τ
using *field-auto-imp-field-hom-on s t* **by** *blast*
have *zE*: $\zeta \in$ *eval-img F* ζ **by** (*rule Subfield.eval-img-self*[*OF sfF*])
— Each automorphism sends ζ to a power of ζ.
have $(\sigma\ \zeta) \hat{} n = 1$ **by** (*rule field-auto-maps-unity-root*[*OF sfF n alg s root*])
then obtain *k l* **where** *sk*: $\sigma\ \zeta = \zeta \hat{} k$ **and** *tl*: $\tau\ \zeta = \zeta \hat{} l$
using *alg field-auto-maps-unity-root n primitive root sfF t* **by** *presburger*
— Powers commute: both composites give $\zeta^{k * l}$.
have $\sigma\ (\tau\ \zeta) = (\sigma\ \zeta) \hat{} l$ **using** *zE*
using *field-hom-on.hom-power homs tl* **by** *auto*
also have ... = $(\zeta \hat{} l) \hat{} k$
by (*metis mult.commute power-mult sk*)
also have ... = $\tau\ (\sigma\ \zeta)$
using *field-hom-on.hom-power homt sk tl zE* **by** *fastforce*
finally show $\sigma\ (\tau\ \zeta) = \tau\ (\sigma\ \zeta)$ **.**
qed

3.4 Kummer extensions are abelian

Kummer case. If F already contains the nth roots of unity and $a^n \in F$, then $F\ a$ has an abelian Galois group. An automorphism sends a to another root of $X^n - a^n$, so $\sigma\ a = a * (\sigma\ a\ /\ a)$ where the ratio $\sigma\ a\ /\ a$ is an nth root of unity, hence lies in F and is fixed by every automorphism. Two such automorphisms therefore commute at a.

theorem *kummer-galois-solvable*:
fixes F :: *complex set* **and** a :: *complex*
assumes *sfF*: *Subfield F* **and** *n*: $n > 0$ **and** *alg*: *algebraic-over F a*
and *anF*: $a \hat{} n \in F$
and *roots-in-F*: $\bigwedge w.\ w \hat{} n = 1 \Longrightarrow w \in F$
and *a-nz*: $a \neq 0$
shows *Group.solvable* (*field-auto* (*eval-img F a*) *F*) (*compose* (*eval-img F a*)) (*identity* (*eval-img F a*))
proof (*rule galois-solvable-of-gen-commute*[*OF sfF alg*])
fix $\sigma\ \tau$ **assume** *s*: $\sigma \in$ *field-auto* (*eval-img F a*) *F* **and** *t*: $\tau \in$ *field-auto* (*eval-img F a*) *F*
let *?E* = *eval-img F a*
have *sfE*: *Subfield ?E* **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])
have *FE*: $F \subseteq$ *?E* **using** *Subfield.eval-img-base*[*OF sfF*] **by** *blast*
have *aE*: $a \in$ *?E* **by** (*rule Subfield.eval-img-self*[*OF sfF*])
obtain *homs*: *field-hom-on ?E* σ **and** *homt*: *field-hom-on ?E* τ
using *field-auto-imp-field-hom-on s sfE t* **by** *blast*
have *fixFs*: $\bigwedge x.\ x \in F \Longrightarrow \sigma\ x = x$ **and** *fixFt*: $\bigwedge x.\ x \in F \Longrightarrow \tau\ x = x$
using *s t* **by** (*auto simp*: *field-auto-mem-iff*)
— An automorphism preserves $X^n - a^n$, so $(\sigma\ a)^n = a^n$; the ratio is a root of unity in F.
define P **where** $P \equiv$ (*monom 1 n* $-$ [:$a \hat{} n$:] :: *complex poly*)
have *PF*: $P \in$ *poly-over F*
unfolding *P-def*
by (*intro Subfield.poly-over-diff*[*OF sfF*] *Subfield.poly-over-monom*[*OF sfF*] *Subfield.poly-over-pCons*[*OF sfF*] *Subfield.poly-over-0*[*OF sfF*] *Subfield.one-closed*[*OF sfF*] *anF*)
have *polyP*: *poly P x* = $x \hat{} n - a \hat{} n$ **for** *x* **by** (*simp add*: *P-def poly-monom*)
have *Proot*: *poly P a* = *0* **by** (*simp add*: *polyP*)
— The ratio $\mu\ a\ /\ a$ for an automorphism μ.
have *ratio*: $\mu\ a\ /\ a \in F \wedge \mu\ a = a * (\mu\ a\ /\ a)$ **if** *m*: $\mu \in$ *field-auto ?E F* **for** μ
using *field-auto-maps-root*[*OF sfE sfF FE m PF aE Proot*]
by (*simp add*: *a-nz polyP power-divide roots-in-F*)
obtain *cs ct* **where** *csF*: $cs \in F$ **and** *sa*: $\sigma\ a = a * cs$ **and** *ctF*: $ct \in F$ **and** *ta*: $\tau\ a = a * ct$
using *ratio s t* **by** *blast*
— The ratios lie in F, so are fixed by both automorphisms; the composites agree at a.
have $\sigma\ (\tau\ a) = \sigma\ a * \sigma\ ct$
using *FE aE ctF field-hom-on.hom-mult homs ta* **by** *fastforce*
also have ... = $(a * cs) * ct$ **by** (*simp add*: *ctF fixFs sa*)
also have ... = $\tau\ (a * cs)$

using *FE aE csF field-hom-on.hom-mult fixFt homt ta* **by** *fastforce*
also have $\ldots = \tau\ (\sigma\ a)$ **by** (*simp add*: *sa*)
finally show $\sigma\ (\tau\ a) = \tau\ (\sigma\ a)$ **.**
qed

end

4 The forward Abel–Ruffini direction: assembling the radical tower

theory *Radical-Solvable*
imports *Cyclotomic-Kummer*
HOL−New-Algebra.Normal-Closure
HOL−New-Algebra.Galois-Restriction-Splitting
HOL−New-Algebra.Radical-Extension
begin

We assemble the pieces — the solvable Kummer/cyclotomic steps, the stability engine and the tower assembly — into "a radical tower over a field containing enough roots of unity has a solvable Galois group".

4.1 A single radical step, with roots of unity present, is a full splitting field

When F contains the nth roots of unity and $a^n \in F$, the simple Kummer extension *eval-img F a* already contains *all* roots of $X^n - a^n$ (the others are $\zeta * a$ for roots of unity $\zeta \in F$), so it *equals* the field generated by that full root set.

lemma *kummer-step-full-root-set*:
fixes F :: *complex set* **and** a :: *complex*
assumes *sfF*: *Subfield F* **and** *n*: $n > 0$ **and** *anF*: $a \mathbin{\hat{}} n \in F$ **and** *a-nz*: $a \neq 0$
and *roots-in-F*: $\bigwedge w.\ w \mathbin{\hat{}} n = 1 \Longrightarrow w \in F$
shows *eval-img F a* = *generate-field* ($F \cup \{r.\ poly\ (monom\ 1\ n - [:a \mathbin{\hat{}} n:])\ r = 0\}$)
proof −
define P **where** $P \equiv (monom\ 1\ n - [:a \mathbin{\hat{}} n:]$ :: *complex poly*)
define X **where** $X \equiv \{r.\ poly\ P\ r = 0\}$
have *polyP*: *poly* $P\ x = x \mathbin{\hat{}} n - a \mathbin{\hat{}} n$ **for** x **by** (*simp add*: *P-def poly-monom*)
— a is algebraic over F: it is a root of the nonzero F-polynomial P.
have *PF*: $P \in$ *poly-over F*
unfolding *P-def*
by (*simp add*: *anF roots-in-F sfF Subfield.poly-over-0 Subfield.poly-over-diff Subfield.poly-over-monom Subfield.poly-over-pCons*)
have *Pnz*: $P \neq 0$
by (*simp add*: *P-def monom-eq-const-iff n*)
have *alg*: *algebraic-over F a*
using *PF Pnz algebraic-over-def polyP* **by** *force*
interpret *E*: *Subfield eval-img F a* **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])

obtain *aE*: $a \in$ *eval-img F a* **and** *FE*: $F \subseteq$ *eval-img F a*
using *Subfield.eval-img-base Subfield.eval-img-self sfF* **by** *blast*
— The other roots of $X^n - a^n$ are $w * a$ with w a root of unity in F.
have *XE*: $X \subseteq$ *eval-img F a*
proof
fix r **assume** $r \in X$
then have *rn*: $r \hat{} n = a \hat{} n$ **by** (*simp add*: *X-def polyP*)
define w **where** $w = r * inverse\ a$
have w: $w \hat{} n = 1$ **using** *a-nz*
by (*metis power-mult-distrib power-one right-inverse rn w-def*)
have $r = w * a$ **using** *a-nz* **by** (*simp add*: *w-def field-simps*)
then show $r \in$ *eval-img F a*
using *w FE aE roots-in-F* **by** *blast*
qed
— Two inclusions between the two generated fields.
have *aX*: $a \in X$ **by** (*simp add*: *X-def polyP*)
have §: *eval-img F a* = *generate-field* $(F \cup \{a\})$
by (*rule Subfield.eval-img-eq-generate-field*[*OF sfF alg*])
have *generate-field* $(F \cup \{a\})$ = *generate-field* $(F \cup X)$
proof
show *generate-field* $(F \cup \{a\}) \subseteq$ *generate-field* $(F \cup X)$
by (*simp add*: *aX generate-field-mono*)
show *generate-field* $(F \cup X) \subseteq$ *generate-field* $(F \cup \{a\})$
by (*metis E.Subfield-axioms FE Un-least XE* § *generate-field-least*)
qed
with § **show** *?thesis* **by** (*simp add*: *P-def X-def*)
qed

4.2 A Kummer step is both solvable and stable

Packaging the previous facts: inside an ambient field K that contains the full root set of $X^n - a^n$, and with the nth roots of unity present in F, the Kummer extension *eval-img F a* supplies *both* ingredients that *solvable-tower* needs for a step: its relative Galois group is solvable (*kummer-galois-solvable*), and it is stable under *field-auto K F* (*root-generated-stable*, since it is a full root-set field by *kummer-step-full-root-set*).

theorem *kummer-step-solvable-and-stable*:
fixes *K F* :: *complex set* **and** *a* :: *complex*
assumes *K*: *Subfield K* **and** *sfF*: *Subfield F* **and** *FK*: $F \subseteq K$
and *n*: $n > 0$ **and** *anF*: $a \hat{} n \in F$ **and** *a-nz*: $a \neq 0$
and *roots-in-F*: $\bigwedge w.\ w \hat{} n = 1 \Longrightarrow w \in F$
and *aK*: $a \in K$
shows *Group.solvable* (*field-auto* (*eval-img F a*) *F*) (*compose* (*eval-img F a*)) (*identity* (*eval-img F a*))
$\wedge$ *eval-img F a* $\subseteq K$
$\wedge$ $(\forall \sigma \in$ *field-auto K F*. σ ‘ *eval-img F a* = *eval-img F a*)
$\wedge$ *Subfield* (*eval-img F a*)
proof −
define *P* **where** $P \equiv$ (*monom 1 n* − [:$a \hat{} n$:])

define X **where** $X \equiv \{r.\ poly\ P\ r = 0\}$
have *polyP*: $poly\ P\ x = x\ \hat{}\ n - a\ \hat{}\ n$ **for** x **by** (*simp add*: *P-def poly-monom*)
have *PF*: $P \in poly\text{-}over\ F$
unfolding *P-def*
by (*metis anF monom-0 sfF Subfield.one-closed Subfield.poly-over-diff Subfield.poly-over-monom*)
have *Pnz*: $P \neq 0$
by (*simp add*: *P-def monom-eq-const-iff n*)
have *alg*: *algebraic-over F a*
unfolding *algebraic-over-def* **using** *PF Pnz* **by** (*auto simp*: *polyP*)
— The Kummer extension equals the full-root-set field of P.
have *E-eq*: *eval-img F a* = *generate-field* $(F \cup X)$
using *kummer-step-full-root-set*[*OF sfF n anF a-nz roots-in-F*] **by** (*simp add*: *P-def X-def*)
have *solv*: *Group.solvable* (*field-auto* (*eval-img F a*) *F*)
(*compose* (*eval-img F a*)) (*identity* (*eval-img F a*))
by (*rule kummer-galois-solvable*[*OF sfF n alg anF roots-in-F a-nz*])
have *finX*: *finite X* **unfolding** *X-def* **using** *Pnz* **by** (*rule poly-roots-finite*)
have *XK*: $X \subseteq K$
proof
fix r **assume** $r \in X$
then have *rn*: $r\ \hat{}\ n = a\ \hat{}\ n$ **by** (*simp add*: *X-def polyP*)
define w **where** $w \equiv r * inverse\ a$
have $w\ \hat{}\ n = 1$
by (*simp add*: *a-nz power-inverse power-mult-distrib rn w-def*)
with *FK* **have** *wK*: $w \in K$
by (*simp add*: *roots-in-F subsetD*)
have $r = w * a$ **using** *a-nz* **by** (*simp add*: *w-def field-simps*)
then show $r \in K$ **using** *wK aK K* **by** (*simp add*: *Subfield.mult-closed*)
qed
have *EK*: *eval-img F a* $\subseteq K$
unfolding *E-eq* **using** *FK XK* **by** (*intro generate-field-least*[*OF K*]) *simp*
— Stability under *field-auto K F*.
have *stable*: σ ‘ *eval-img F a* = *eval-img F a* **if** *s*: $\sigma \in$ *field-auto K F* **for** σ
unfolding *E-eq*
by (*metis FK K PF XK X-def finX root-generated-stable sfF that*)
have *sfE*: *Subfield* (*eval-img F a*) **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])
show *?thesis* **using** *solv EK stable sfE* **by** *blast*
qed

4.3 A cyclotomic step is solvable, stable, and supplies the roots of unity

Adjoining the primitive nth root of unity *unity-root n* to F: the extension *eval-img F* (*unity-root n*) has a solvable (abelian) Galois group (*cyclotomic-galois-solvable*), is the full root-set field of $X^n - 1$ (hence stable by *root-generated-stable*), and — crucially — *contains all the nth roots of unity*, establishing the invariant that the subsequent Kummer step needs.

theorem *cyclotomic-step-solvable-and-stable*:
fixes K F :: *complex set*
assumes K: *Subfield* K **and** *sfF*: *Subfield* F **and** FK: $F \subseteq K$ **and** n: $n > 0$
and zK: *unity-root* $n \in K$
shows *Group.solvable* (*field-auto* (*eval-img* F (*unity-root* n)) F)
(*compose* (*eval-img* F (*unity-root* n))) (*identity* (*eval-img* F (*unity-root* n)))
$\wedge$ *eval-img* F (*unity-root* n) $\subseteq K$
$\wedge$ ($\forall \sigma \in$ *field-auto* K F. σ ‘ *eval-img* F (*unity-root* n) = *eval-img* F (*unity-root* n))
$\wedge$ *Subfield* (*eval-img* F (*unity-root* n))
$\wedge$ ($\forall w$. $w \mathbin{\hat{}} n = 1 \longrightarrow w \in$ *eval-img* F (*unity-root* n))
proof −
define ζ **where** ζ = *unity-root* n
have *zroot*: $\zeta \mathbin{\hat{}} n = 1$ **unfolding** ζ-*def* **using** *unity-root-mem*[*OF* n] **by** *simp*
— Every nth root of unity is a power of ζ (primitivity).
have *primitive*: $\exists k$. $w = \zeta \mathbin{\hat{}} k$ **if** $w \mathbin{\hat{}} n = 1$ **for** w
using ζ-*def* n *roots-of-unity-subset-powers that* **by** *auto*
define P **where** $P \equiv$ (*monom* 1 n − 1 :: *complex poly*)
define X **where** $X \equiv \{r.\ poly\ P\ r = 0\}$
have *polyP*: *poly* P $x = x \mathbin{\hat{}} n - 1$ **for** x **by** (*simp add*: *P-def poly-monom*)
have *PF*: $P \in$ *poly-over* F
by (*simp add*: *P-def sfF Subfield.one-closed Subfield.poly-over-1 Subfield.poly-over-diff Subfield.poly-over-monom*)
have *Pnz*: $P \neq 0$
by (*metis n polyP poly-0 right-minus-eq zero-neq-one zero-power*)
have *Xroots*: X = *roots-of-unity* n **by** (*auto simp*: *X-def polyP roots-of-unity-def*)
have *alg*: *algebraic-over* F ζ
unfolding *algebraic-over-def* **using** *PF Pnz zroot* **by** (*auto simp*: *polyP*)
interpret E: *Subfield eval-img* F ζ **by** (*rule Subfield.subfield-eval-img*[*OF sfF alg*])
— All roots of unity are powers of ζ, hence lie in *eval-img* F ζ.
have *rootsE*: $w \in$ *eval-img* F ζ **if** *wn*: $w \mathbin{\hat{}} n = 1$ **for** w
using *primitive wn sfF Subfield.eval-img-self* **by** *blast*
— *eval-img* F ζ equals the full root-set field of P.
have XE: $X \subseteq$ *eval-img* F ζ **using** *Xroots rootsE* **by** (*auto simp*: *roots-of-unity-def*)
have *E-eq*: *eval-img* F ζ = *generate-field* ($F \cup X$)
proof −
have §: *eval-img* F ζ = *generate-field* ($F \cup \{\zeta\}$)
by (*rule Subfield.eval-img-eq-generate-field*[*OF sfF alg*])
moreover have *generate-field* ($F \cup \{\zeta\}$) = *generate-field* ($F \cup X$)
proof
have $F \cup \{\zeta\} \subseteq F \cup X$
by (*simp add*: *Xroots zroot*)
then show *generate-field* ($F \cup \{\zeta\}$) $\subseteq$ *generate-field* ($F \cup X$) **by** (*rule generate-field-mono*)
have $F \cup X \subseteq$ *eval-img* F ζ
using *XE sfF Subfield.eval-img-base* **by** *auto*
then show *generate-field* ($F \cup X$) $\subseteq$ *generate-field* ($F \cup \{\zeta\}$)
by (*metis* § *subfield-generate-field generate-field-least*)
qed
ultimately show *?thesis* **by** *simp*

```
  qed
  — Containment in K: roots of unity are powers of ζ ∈ K.
  have XK: X ⊆ K
    using K primitive Subfield.power-closed zK by (force simp: ζ-def Xroots)
  have EK: eval-img F ζ ⊆ K
    unfolding E-eq using FK XK by (intro generate-field-least[OF K]) simp
  have finX: finite X
    by (simp add: Pnz X-def poly-roots-finite)
  have stable: σ ‘ eval-img F ζ = eval-img F ζ if s: σ ∈ field-auto K F for σ
    unfolding E-eq
    using root-generated-stable[OF K sfF FK PF] XK X-def finX that by blast
  show ?thesis
    using cyclotomic-galois-solvable[OF sfF n alg zroot primitive]
    using EK stable E.Subfield-axioms rootsE by (simp add: ζ-def)
qed
```

4.4 A chain of Kummer steps is a solvable tower

An element whose nth power lies in F is algebraic over F (it is a root of the nonzero F-polynomial $X^n - a^n$).

```
lemma radical-algebraic:
  fixes F :: complex set
  assumes sfF: Subfield F and n: n > 0 and anF: a ^ n ∈ F
  shows algebraic-over F a
proof −
  define P where P ≡ (monom 1 n − [:a ^ n:] :: complex poly)
  have polyP: poly P x = x ^ n − a ^ n for x by (simp add: P-def poly-monom)
  have PF: P ∈ poly-over F
   by (simp add: P-def anF sfF Subfield.one-closed Subfield.poly-over-0 Subfield.poly-over-diff
       Subfield.poly-over-monom Subfield.poly-over-pCons)
  have Pnz: P ≠ 0
    by (metis n polyP poly-0 power-one right-minus-eq zero-neq-one zero-power)
  show ?thesis unfolding algebraic-over-def using PF Pnz by (auto simp: polyP)
qed
```

The chain of fields built from a list of *Kummer witnesses* (a, n) (each contributing a step $F \rightsquigarrow F(a)$).

```
fun kummer-fields :: complex set ⇒ (complex × nat) list ⇒ complex set list where
  kummer-fields F [] = []
| kummer-fields F ((a, n) # ws) = eval-img F a # kummer-fields (eval-img F a) ws
```

Validity of a witness chain over F: each step has positive exponent, a nonzero base a with $a^n \in F$, and the nth roots of unity already present in the current field.

```
fun kummer-valid :: complex set ⇒ (complex × nat) list ⇒ bool where
  kummer-valid F [] = True
| kummer-valid F ((a, n) # ws) =
    (n > 0 ∧ a ≠ 0 ∧ a ^ n ∈ F ∧ (∀ w. w ^ n = 1 ⟶ w ∈ F) ∧
```

kummer-valid (*eval-img F a*) *ws*)

The top field of the chain (the ambient splitting field).

definition *ktop* :: *complex set* ⇒ (*complex* × *nat*) *list* ⇒ *complex set* **where**
ktop F ws ≡ *last* (*F* # *kummer-fields F ws*)

lemma *ktop-Nil* [*simp*]: *ktop F* [] = *F*
by (*simp add*: *ktop-def*)

lemma *ktop-Cons* [*simp*]: *ktop F* ((*a*, *n*) # *ws*) = *ktop* (*eval-img F a*) *ws*
by (*simp add*: *ktop-def*)

The chain is increasing, so the base is contained in the top.

lemma *ktop-facts*:
fixes *F* :: *complex set*
shows *Subfield F* ⟹ *kummer-valid F ws* ⟹ *F* ⊆ *ktop F ws* ∧ *Subfield* (*ktop F ws*)
proof (*induction ws arbitrary*: *F*)
case (*Cons w ws*)
then obtain *a n* **where** *an*: *w* = (*a*, *n*) **and** *n* > *0 a* ^ *n* ∈ *F* **and** *valid′*: *kummer-valid* (*eval-img F a*) *ws*
by (*cases w*) *auto*
then have *Subfield* (*eval-img F a*)
using *Cons.prems radical-algebraic* **by** (*simp add*: *Subfield.subfield-eval-img*)
then show *?case* **using** *an Cons*
by (*simp add*: *Subfield.eval-img-base subset-iff*)
qed *auto*

The top field is a subfield.

lemma
assumes *Subfield F kummer-valid F ws*
shows *F-subset-ktop*: *F* ⊆ *ktop F ws* **and** *subfield-ktop*: *Subfield* (*ktop F ws*)
using *ktop-facts* [*OF assms*] **by** *auto*

The chain is a solvable tower. By induction on the witness list: each Kummer step is solvable and stable inside the ambient top field.

theorem *kummer-chain-solvable-tower*:
Subfield F ⟹ *kummer-valid F ws* ⟹ *solvable-tower* (*ktop F ws*) *F* (*kummer-fields F ws*)
proof (*induction ws arbitrary*: *F*)
case *Nil*
then show *?case* **by** *simp*
next
case (*Cons w ws*)
then obtain *a n* **where** *an*: *w* = (*a*, *n*) **and** *n*: *n* > *0* **and** *anF*: *a* ^ *n* ∈ *F* **and** *a-nz*: *a* ≠ *0*
and *rou*: ⋀*w*. *w* ^ *n* = *1* ⟹ *w* ∈ *F* **and** *valid′*: *kummer-valid* (*eval-img F a*) *ws*
by (*cases w*) *auto*
let *?G* = *eval-img F a*

```
  have sfF: Subfield F by (rule Cons.prems(1))
  have sfG: Subfield ?G
    using anF n radical-algebraic sfF Subfield.subfield-eval-img by blast
  define K where K ≡ ktop ?G ws
  have sfK: Subfield K unfolding K-def using subfield-ktop[OF sfG valid'] .
  have GK: ?G ⊆ K unfolding K-def using F-subset-ktop[OF sfG valid'] .
  have FG: F ⊆ ?G using Subfield.eval-img-base[OF sfF] by blast
  have FK: F ⊆ K using FG GK by blast
  have aK: a ∈ K using Subfield.eval-img-self[OF sfF] GK by blast
  — The rest of the chain is a solvable tower, by the induction hypothesis.
  have solvable-tower K ?G (kummer-fields ?G ws)
    using Cons.IH[OF sfG valid'] unfolding K-def .
  — The step is solvable and stable inside K.
  then have solvable-tower K F (?G # kummer-fields ?G ws)
    using kummer-step-solvable-and-stable[OF sfK sfF FK n anF a-nz rou aK]
    by (simp add: FG sfF)
  then show ?case using an by (simp add: K-def)
qed
```

The Galois group of a valid Kummer chain is solvable.

```
corollary kummer-chain-galois-solvable:
  assumes sfF: Subfield F and valid: kummer-valid F ws
  shows Group.solvable (field-auto (ktop F ws) F) (compose (ktop F ws)) (identity
(ktop F ws))
 using kummer-chain-solvable-tower sfF solvable-tower-imp-solvable subfield-ktop valid
  by blast
```

4.5 Solvability descends to a stable subfield with surjective restriction

Descent to an intermediate Galois group. Let $F \subseteq E \subseteq K$ with E stable under *field-auto K F*, and suppose restriction $\lambda\sigma.\ \mathit{restrict}\ \sigma\ E$ maps *field-auto K F* *onto* *field-auto E F* (the *normality* input: every F-automorphism of E extends to one of K). Then if *field-auto K F* is solvable, so is *field-auto E F* — it is a homomorphic image.

This is the group-theoretic heart of the forward direction: the quintic's Galois group would be a *quotient* of the solvable Galois group of a radical extension, hence solvable — contradicting its non-solvability. The surjectivity hypothesis is exactly what *normality of K over F* supplies (and what a full normal-closure construction would discharge).

```
theorem galois-solvable-descends:
  fixes K E F :: complex set
  assumes K: Subfield K and sfE: Subfield E and sfF: Subfield F
    and FE: F ⊆ E and EK: E ⊆ K
    and stable: ⋀σ. σ ∈ field-auto K F ⟹ σ ‘ E = E
    and onto: (λσ. restrict σ E) ‘ field-auto K F = field-auto E F
    and solvK: Group.solvable (field-auto K F) (compose K) (identity K)
  shows Group.solvable (field-auto E F) (compose E) (identity E)
```

proof −
— Restriction is a group homomorphism onto *field-auto E F* (stability + surjectivity).
interpret *H*: *group-homomorphism galois-restriction K E F*
field-auto K F compose K identity K field-auto E F compose E identity E
by (*simp add*: *EK FE K galois-restriction-def galois-restriction-hom sfE stable*)
have *surj*: *galois-restriction K E F ‘ field-auto K F = field-auto E F*
using *onto* **by** (*simp add*: *galois-restriction-def*)
interpret *Epi*: *group-epimorphism galois-restriction K E F*
field-auto K F compose K identity K field-auto E F compose E identity E
by (*rule galois-restriction-epimorphism*[*OF K sfE FE EK stable surj*])
show *?thesis* **by** (*rule Epi.solvable-image-epi*[*OF solvK*])
qed

4.6 The forward theorem

The forward Abel–Ruffini theorem. Suppose a set S lies in a field E that sits inside the top *ktop F ws* of a valid Kummer chain over F (so E is contained in a radical extension), with E stable under the ambient Galois group and restriction onto *field-auto E F* surjective (the normality conditions). Then *field-auto E F* is solvable.

Chaining *kummer-chain-galois-solvable* (the radical top has a solvable Galois group) with *galois-solvable-descends* (solvability descends to the stable subfield). This is the complete forward direction modulo the single classical input — normality of the ambient radical field, i.e. surjectivity of restriction — which a full normal-closure construction supplies.

theorem *radical-extension-galois-solvable*:
fixes *F E* :: *complex set* **and** *ws* :: (*complex* × *nat*) *list*
assumes *sfF*: *Subfield F* **and** *sfE*: *Subfield E*
and *valid*: *kummer-valid F ws*
and *FE*: $F \subseteq E$ **and** *EK*: $E \subseteq$ *ktop F ws*
and *stable*: $\bigwedge\sigma.\ \sigma \in$ *field-auto* (*ktop F ws*) $F \Longrightarrow \sigma$ ‘ $E = E$
and *onto*: $(\lambda\sigma.$ *restrict* $\sigma\ E)$ ‘ *field-auto* (*ktop F ws*) $F =$ *field-auto E F*
shows *Group.solvable* (*field-auto E F*) (*compose E*) (*identity E*)
proof −
have *sfK*: *Subfield* (*ktop F ws*) **by** (*rule subfield-ktop*[*OF sfF valid*])
show *?thesis*
using *galois-solvable-descends*[*OF sfK sfE sfF FE EK stable onto*]
using *kummer-chain-galois-solvable sfF valid* **by** *blast*
qed

4.7 From a roots-of-unity-free chain to a valid Kummer chain

A *pre-Kummer chain* over F bounded by N: each step has positive exponent n *dividing* N, a nonzero base a with $a^n \in F$. This is the roots-of-unity-*free* data of a radical tower (a raw radical step, tagged with its exponent), with the mild global bound that all exponents divide a common N.

fun *pre-kummer-valid* :: *nat* ⇒ *complex set* ⇒ (*complex* × *nat*) *list* ⇒ *bool* **where**
  pre-kummer-valid N F [] = *True*
| *pre-kummer-valid N F* ((*a*, *n*) # *ws*) =
    (*n* > *0* ∧ *n dvd N* ∧ *a* ≠ *0* ∧ *a* ^ *n* ∈ *F* ∧ *pre-kummer-valid N* (*eval-img F a*) *ws*)

If the base field already contains all Nth roots of unity, a pre-Kummer chain bounded by N is a genuine Kummer chain: at each step the nth roots of unity (with *n dvd N*) are among the Nth roots of unity (*roots-of-unity-dvd-mono*), which persist up the tower since each field contains its predecessor.

lemma *pre-kummer-valid-imp-kummer-valid*:
  Subfield F ⟹ (∀ *w*. *w* ^ *N* = *1* ⟶ *w* ∈ *F*) ⟹ *pre-kummer-valid N F ws* ⟹ *kummer-valid F ws*
proof (*induction ws arbitrary*: *F*)
  case *Nil*
  then show *?case* **by** *simp*
next
  case (*Cons w ws*)
  then obtain *a n* **where** *an*: *w* = (*a*, *n*)
  and *n* > *0* **and** *ndvd*: *n dvd N* **and** *a-nz*: *a* ≠ *0* **and** *anF*: *a* ^ *n* ∈ *F* **and** *pre′*: *pre-kummer-valid N* (*eval-img F a*) *ws*
    by (*metis pre-kummer-valid.simps*(*2*) *surj-pair*)
  — Each nth root of unity (*n dvd N*) is an Nth root of unity, hence in F.
  have *rou-n*: ⋀*v*. *v* ^ *n* = *1* ⟹ *v* ∈ *F*
    using *ndvd roots-of-unity-dvd-mono Cons.prems* **by** *blast*
  have *sfG*: *Subfield* (*eval-img F a*)
    using ‹*Subfield F*› *anF* ‹*n* > *0*› *radical-algebraic Subfield.subfield-eval-img* **by** *blast*
  have *FG*: *F* ⊆ *eval-img F a* **using** *Subfield.eval-img-base*[*OF Cons.prems*(*1*)] **by** *blast*
  — The roots-of-unity invariant persists to the next field (it contains F).
  have *kummer-valid* (*eval-img F a*) *ws*
    using *Cons FG pre′ sfG* **by** *blast*
  then show *?case* **using** ‹*n* > *0*› *a-nz anF rou-n an* **by** *simp*
qed

The Galois group of a pre-Kummer chain, once the roots of unity are adjoined, is solvable. If F contains the Nth roots of unity and *ws* is a pre-Kummer chain bounded by N, then *field-auto* (*ktop F ws*) F is solvable.

corollary *pre-kummer-chain-galois-solvable*:
  assumes *sfF*: *Subfield F* **and** *rou*: ∀ *w*. *w* ^ *N* = *1* ⟶ *w* ∈ *F*
    and *pre*: *pre-kummer-valid N F ws*
  shows *Group.solvable* (*field-auto* (*ktop F ws*) *F*) (*compose* (*ktop F ws*)) (*identity* (*ktop F ws*))
  using *kummer-chain-galois-solvable*[*OF sfF pre-kummer-valid-imp-kummer-valid*[*OF sfF rou pre*]] **.**

4.8 Extracting Kummer witnesses from a radical tower

A *nondegenerate* radical tower is one whose every step can be realised by a *nonzero* radical whose exponent divides a common bound N. (Adjoining 0 adds nothing, so this is no real restriction for reaching a target set; and any finite family of exponents divides their common multiple.) We record this as the existence of a witness list.

definition *realises-tower* :: *nat* ⇒ *complex set* ⇒ (*complex* × *nat*) *list* ⇒ *complex set list* ⇒ *bool*
  where *realises-tower N F ws Fs* ≡ *pre-kummer-valid N F ws* ∧ *kummer-fields F ws* = *Fs*

If a radical tower is realised by a nondegenerate witness list over a base F containing the Nth roots of unity, then its top field has a solvable Galois group over F. The top field of the realised chain coincides with the tower top.

lemma *ktop-eq-tower-top*:
  assumes *kummer-fields F ws* = *Fs*
  shows *ktop F ws* = *tower-top F Fs*
  using *assms* **by** (*simp add*: *ktop-def tower-top-def*)

theorem *radical-tower-galois-solvable*:
  assumes *sfF*: *Subfield F* **and** *rou*: ∀ *w*. *w* ^ *N* = *1* ⟶ *w* ∈ *F*
    and *real*: *realises-tower N F ws Fs*
  shows *Group.solvable* (*field-auto* (*tower-top F Fs*) *F*)
        (*compose* (*tower-top F Fs*)) (*identity* (*tower-top F Fs*))
proof −
  have *pre*: *pre-kummer-valid N F ws* **and** *kf*: *kummer-fields F ws* = *Fs*
    using *real* **by** (*auto simp*: *realises-tower-def*)
  then show *?thesis* **using** *ktop-eq-tower-top*[*OF kf*]
    by (*metis pre-kummer-chain-galois-solvable rou sfF*)
qed

Conversely, a nondegenerate witness chain *is* a radical tower in the sense of *radical-tower*: each Kummer step $F \rightsquigarrow F(a)$ is a radical step, since *eval-img F a* = *generate-field* $(F \cup \{a\})$ with $a^n \in F$. Thus the notion of a realised tower is faithful to the project's *radical-tower*.

4.9 Extracting a nondegenerate witness list from a bare radical tower

The nondegeneracy conditions of a witness list, without the common divisibility bound: each step has positive exponent, a nonzero base, and its power in the current field.

fun *wit-valid* :: *complex set* ⇒ (*complex* × *nat*) *list* ⇒ *bool* **where**
  wit-valid F [] = *True*
| *wit-valid F* ((*a*, *n*) # *ws*) =
    (*n* > *0* ∧ *a* ≠ *0* ∧ *a* ^ *n* ∈ *F* ∧ *wit-valid* (*eval-img F a*) *ws*)

Given a common bound N divisible by every step exponent, a nondegenerate witness list is a pre-Kummer chain.

lemma *wit-valid-imp-pre-kummer-valid*:
  wit-valid F ws $\Longrightarrow$ ($\forall$ *x* $\in$ *set ws. snd x dvd N*) $\Longrightarrow$ *pre-kummer-valid N F ws*
proof (*induction ws arbitrary*: *F*)
  case *Nil*
  then show *?case* **by** *simp*
next
  case (*Cons w ws*)
  then show *?case* **by** (*cases w*) *force*
qed

Every radical tower has a nondegenerate witness list. Walking the tower: each radical step $F \rightsquigarrow G$ either uses a nonzero radical ($G =$ *eval-img F a*) or is a no-op ($G = F$, which occurs when the existential witness is *0*); in the latter case the trivial witness (*1*, *1*) reproduces F (as *eval-img F 1* $= F$).

lemma *radical-tower-witnesses*:
  Subfield F $\Longrightarrow$ *radical-tower F Fs* $\Longrightarrow$ $\exists$ *ws. wit-valid F ws* $\wedge$ *kummer-fields F ws* = *Fs*
proof (*induction Fs arbitrary*: *F*)
  case *Nil*
  show *?case* **by** (*rule exI*[*of* - []]) *simp*
next
  case (*Cons G Gs*)
  interpret *F*: *Subfield F* **by** (*rule* ‹*Subfield F*›)
  have *step*: *radical-step F G* **and** *tower'*: *radical-tower G Gs* **using** *Cons.prems*(*2*)
by *auto*
  obtain *a n* **where** *n* > *0* **and** *anF*: *a* ^ *n* $\in$ *F* **and** *Gdef*: *G* = *generate-field* (*F* $\cup$ {*a*})
    using *step* **by** (*auto simp*: *radical-step-def*)
  — Either the step is a genuine nonzero adjunction, or a no-op producing F again.
  consider (*nz*) *a* $\neq$ *0* | (*z*) *a* = *0* **by** *blast*
  then show *?case*
  proof *cases*
    case *nz*
    have *sfG*: *Subfield G*
      using *local.step radical-step-field* **by** *auto*
    obtain *ws* **where** *wit-valid G ws kummer-fields G ws* = *Gs*
      using *Cons.IH*[*OF sfG tower'*] **by** *blast*
    then show *?thesis*
      using *anF kummer-fields.simps*(*2*) ‹*n* > *0*› *nz wit-valid.simps*(*2*)
      by (*metis F.eval-img-eq-generate-field F.Subfield-axioms Gdef radical-algebraic*)
  next
    case *z*
    have *F* $\cup$ {*a*} = *F* **using** *z F.zero-closed* **by** *auto*
    then have *GF*: *G* = *F* **using** *Gdef* **by** (*simp add*: *F.generate-field-self*)
    — The trivial witness (*1*, *1*) keeps the field at F.
     have *alg1*: *algebraic-over F 1* **using** *radical-algebraic*[*OF* ‹*Subfield F*›, *of 1 1*]
F.one-closed **by** *simp*

```
    have e1: eval-img F 1 = F
      using Subfield.eval-img-eq-generate-field[OF ‹Subfield F› alg1] F.one-closed
      by (simp add: insert-absorb F.generate-field-self)
    obtain ws where ws: wit-valid F ws kummer-fields F ws = Gs
      using Cons.IH F.Subfield-axioms GF tower' by blast
    have wit-valid F ((1, 1) # ws) ∧ kummer-fields F ((1, 1) # ws) = G # Gs
      using ws e1 F.one-closed GF by simp
    then show ?thesis by blast
  qed
qed

lemma pre-kummer-valid-imp-radical-tower:
  Subfield F ⟹ pre-kummer-valid N F ws ⟹ radical-tower F (kummer-fields F ws)
proof (induction ws arbitrary: F)
  case Nil
  then show ?case by simp
next
  case (Cons w ws)
  then obtain a n where an: w = (a, n) and n > 0 and anF: a ^ n ∈ F
    by (metis pre-kummer-valid.simps(2) surj-pair)
  have alg: algebraic-over F a using radical-algebraic[OF Cons.prems(1) ‹n > 0›
anF] .
  — The first step is a radical step.
  have eval-img F a = generate-field (F ∪ {a})
    by (rule Subfield.eval-img-eq-generate-field[OF Cons.prems(1) alg])
  then have radical-step F (eval-img F a)
    unfolding radical-step-def using ‹n > 0› anF by blast
  — The rest is a radical tower, by the induction hypothesis.
  then show ?case using an Cons
    using radical-step-field by auto
qed
```

4.10 A radical tower over a roots-of-unity base has a solvable Galois group

The forward theorem for a bare radical tower. If F contains *enough* roots of unity — namely the Nth roots of unity for N the product of the (extracted) step exponents — then the top of any radical tower over F has a solvable Galois group over F. This reduces the forward direction over F to the single remaining hypothesis that F contains the required roots of unity — which a cyclotomic pre-step $F \rightsquigarrow F(\zeta_N)$ supplies (and which, over $\mathbb{Q}$, is the only reason the base must first be enlarged).

```
theorem radical-tower-solvable-of-roots:
  assumes sfF: Subfield F and tower: radical-tower F Fs
    and rou: ⋀ws. wit-valid F ws ⟹ kummer-fields F ws = Fs
          ⟹ ∀ w. w ^ (prod-list (List.map snd ws)) = 1 ⟶ w ∈ F
  shows Group.solvable (field-auto (tower-top F Fs) F)
        (compose (tower-top F Fs)) (identity (tower-top F Fs))
```

proof −
  obtain *ws* **where** *wv*: *wit-valid F ws* **and** *kf*: *kummer-fields F ws = Fs*
    using *radical-tower-witnesses*[*OF sfF tower*] **by** *blast*
  define *N* **where** *N ≡ prod-list* (*List.map snd ws*)
  — Every step exponent divides *N*.
  have *dvdN*: ∀ *x ∈ set ws. snd x dvd N*
    by (*simp add*: *N-def prod-list-dvd*)
  have *pre*: *pre-kummer-valid N F ws* **by** (*rule wit-valid-imp-pre-kummer-valid*[*OF wv dvdN*])
  have *real*: *realises-tower N F ws Fs* **using** *pre kf* **by** (*simp add*: *realises-tower-def*)
  have *rouN*: ∀ *w. w* ^ *N = 1* ⟶ *w ∈ F* **using** *rou*[*OF wv kf*] **by** (*simp add*: *N-def*)
  show *?thesis* **by** (*rule radical-tower-galois-solvable*[*OF sfF rouN real*])
qed

4.11 Unconditional descent to a stable splitting subfield

Combining the splitting-field image theorem *galois-restriction-image-of-splitting-field* with the descent theorem *galois-solvable-descends* discharges the former surjectivity obligation when K is a splitting field over E. This is where the "deep gap" closes.

theorem *galois-splitting-subfield-solvable*:
  fixes *K E F* :: *complex set* **and** *Phi* :: *complex poly*
  assumes *sfE*: *Subfield E* **and** *sfF*: *Subfield F* **and** *FE*: *F ⊆ E*
    and *Phi*: *Phi ∈ poly-over F*
    and *split*: *splitting-field E Phi K*
    and *stable*: ⋀*σ. σ ∈ field-auto K F* ⟹ *σ* ‘ *E = E*
    and *solvK*: *Group.solvable* (*field-auto K F*) (*compose K*) (*identity K*)
  shows *Group.solvable* (*field-auto E F*) (*compose E*) (*identity E*)
proof −
  have *K*: *Subfield K*
    using *splitting-field-subfield*[*OF split*]
    by (*simp add*: *complex-subfield-iff-subfield*)
  have *EK*: *E ⊆ K*
    by (*rule splitting-field-base-subset*[*OF split*])
  — The splitting-field restriction theorem supplies surjectivity, including both image inclusions and the finite extension argument.
  have *onto-named*: *galois-restriction K E F* ‘ *field-auto K F = field-auto E F*
    by (*rule galois-restriction-image-of-splitting-field*[*OF split sfE sfF FE Phi stable*])
  have *onto*: (*λσ. restrict σ E*) ‘ *field-auto K F = field-auto E F*
    using *onto-named* **by** (*simp add*: *galois-restriction-def*)
  show *?thesis*
    by (*rule galois-solvable-descends*[*OF K sfE sfF FE EK stable onto solvK*])
qed

end

5 The normal radical closure of a radical tower

theory *Radical-Normal-Closure*
  imports *Radical-Solvable*
    HOL−New-Algebra.Algebraic-Transitivity
    HOL−New-Algebra.Galois-Transitivity
begin

To descend solvability of a radical extension onto a *quotient* Galois group we must first replace the radical tower by one whose top is *normal* over the base — the "normal radical closure". The construction (as in the standard textbook argument, and in Mathlib's *FieldTheory.AbelRuffini*): the conjugates of a radical tower are again radical towers (*radical-tower-image*), and splicing the finitely many conjugates together (*radical-tower-spliced*) yields one radical tower whose top is stable under the ambient Galois group. These blocks develop the conjugate/splice reasoning; the quintic-specific assembly and the contradiction live in the joint session theory.

5.1 A Galois conjugate of a radical tower is a radical tower

lemma *radical-tower-conjugate*:
  fixes *N F* :: *complex set* **and** *Fs* :: *complex set list* **and** *σ* :: *complex* ⇒ *complex*
  assumes *N*: *Subfield N* **and** *sfF*: *Subfield F*
    and *tower*: *radical-tower F Fs* **and** *topN*: *tower-top F Fs* ⊆ *N*
    and *s*: *σ* ∈ *field-auto N F*
  shows *radical-tower F* (*List.map* ((‘) *σ*) *Fs*)
proof −
  have *field-hom-on N σ*
    using *N s* **by** (*rule field-auto-imp-field-hom-on*)
  then have *image-tower*: *radical-tower* (*σ* ‘ *F*) (*List.map* ((‘) *σ*) *Fs*)
    using *N radical-tower-image sfF topN tower* **by** *blast*
  then show *?thesis* **using** *image-tower*
    by (*metis field-auto-image-base s*)
qed

5.2 The family of all conjugates is a family of towers over the base

Applying a list *σs* of *F*-automorphisms to a radical tower yields a family of radical towers over *F* — the input shape that *radical-tower-spliced* consumes.

lemma *conjugate-towers-over*:
  fixes *N F* :: *complex set* **and** *Fs* :: *complex set list*
    and *σs* :: (*complex* ⇒ *complex*) *list*
  assumes *N*: *Subfield N* **and** *sfF*: *Subfield F*
    and *tower*: *radical-tower F Fs* **and** *topN*: *tower-top F Fs* ⊆ *N*
    and *ss*: *set σs* ⊆ *field-auto N F*
  shows *all-towers-over F* (*List.map* (*λσ*. *List.map* ((‘) *σ*) *Fs*) *σs*)
  using *ss*

proof (*induction* σs)
  case (*Cons* σ σs)
  then show *?case*
    using *N radical-tower-conjugate sfF topN tower* **by** *force*
qed *auto*

5.3 The spliced tower's top contains every tower of the family

Base-changing a tower *Ts* (over F) onto a larger base B can only enlarge its top.

lemma *tower-top-basechange-superset*:
  assumes $F \subseteq B$
  shows *tower-top F Ts* $\subseteq$ *tower-top B* (*List.map* (λH. *generate-field* ($B \cup H$)) *Ts*)
  using *assms*
  by (*auto simp*: *subset-generate-field tower-top-def generate-field.generate-field-base last-map*)

In the spliced tower *spliced B Tss* over B, the top field contains the top of *every* tower of the family (measured over the common base F). This is what makes the splice top a compositum of the conjugates.

lemma *spliced-top-contains-towers*:
  assumes *Subfield B* **and** *Subfield F* **and** $F \subseteq B$ **and** *all-towers-over F Tss*
  shows $\forall$ *Ts* $\in$ *set Tss*. *tower-top F Ts* $\subseteq$ *tower-top B* (*spliced B Tss*)
  using *assms*
proof (*induction Tss arbitrary*: *B*)
  case *Nil*
  show *?case* **by** *simp*
next
  case (*Cons Ts Tss*)
  define Ts' **where** $Ts' \equiv$ *List.map* (λH. *generate-field* ($B \cup H$)) *Ts*
  define B' **where** $B' \equiv$ *tower-top B Ts'*
  — The head tower base-changes onto B, giving a radical tower over B with top B'.
  have *radical-tower B Ts'* **and** *rest*: *all-towers-over F Tss*
    using *Cons.prems Ts'-def radical-tower-basechange-subfield* **by** *auto*
  then obtain *sfB'*: *Subfield B'* **and** *BB'*: $B \subseteq B'$
    by (*simp add*: *B'-def Cons.prems*(*1*) *subfield-tower-top*)
  have *FB'*: $F \subseteq B'$ **using** *Cons.prems*(*3*) *BB'* **by** *blast*
  — The full splice over B equals the head Ts' followed by the splice over B'.
  have *spliced B* (*Ts* # *Tss*) = *Ts'* @ *spliced B' Tss*
    by (*simp add*: *Ts'-def B'-def Let-def*)
  then have *top-eq*: *tower-top B* (*spliced B* (*Ts* # *Tss*)) = *tower-top B'* (*spliced B' Tss*)
    by (*simp add*: *tower-top-append B'-def*)
  — The remainder is a radical tower over B'; hence B' lies below the final top.
  have *B'top*: $B' \subseteq$ *tower-top B'* (*spliced B' Tss*)
    using *FB'* ‹*Subfield F*› *radical-tower-spliced Cons.prems*(*4*) *sfB' subfield-tower-top*
by *force*

```
  show ?case
  proof
    fix Ts2 assume Ts2 ∈ set (Ts # Tss)
    then consider Ts2 = Ts | Ts2 ∈ set Tss by auto
    then show tower-top F Ts2 ⊆ tower-top B (spliced B (Ts # Tss))
      using Cons.IH[OF sfB' Cons.prems(2) FB']
      by (metis B'-def B'top Cons.prems(3) Ts'-def order.trans rest top-eq
          tower-top-basechange-superset)
  qed
qed
```

5.4 The normal radical closure exists

The top of a conjugated tower is the conjugate of the top: since σ fixes F setwise, *tower-top F (map ((‘) σ) Fs) = σ ‘ tower-top F Fs* (the last field of the mapped list is the image of the last field; for the empty tower both sides are σ ‘ $F = F$).

```
lemma tower-top-conjugate:
  assumes sF: σ ‘ F = F
  shows tower-top F (List.map ((‘) σ) Fs) = σ ‘ tower-top F Fs
  by (simp add: last-map sF tower-top-def)
```

The normal radical closure exists (complex form). Given a radical tower Fs over F inside an ambient field N, and any list σs of F-automorphisms of N, splicing the conjugate family yields a single radical tower over F whose top B contains the conjugate σ ‘ *tower-top F Fs* of the original top for every $\sigma \in$ *set* σs.

```
theorem radical-normal-closure-exists:
  fixes N F :: complex set and Fs :: complex set list
    and σs :: (complex ⇒ complex) list
  assumes N: Subfield N and sfF: Subfield F
    and tower: radical-tower F Fs and topN: tower-top F Fs ⊆ N
    and ss: set σs ⊆ field-auto N F
  defines Tss ≡ List.map (λσ. List.map ((‘) σ) Fs) σs
  shows radical-tower F (spliced F Tss)
      (∀σ ∈ set σs. σ ‘ tower-top F Fs ⊆ tower-top F (spliced F Tss))
proof −
  have allt: all-towers-over F Tss
    unfolding Tss-def by (rule conjugate-towers-over[OF N sfF tower topN ss])
  show rt: radical-tower F (spliced F Tss)
    by (rule radical-tower-spliced[OF sfF sfF subset-refl allt])
  show ∀σ ∈ set σs. σ ‘ tower-top F Fs ⊆ tower-top F (spliced F Tss)
  proof
    fix σ assume s: σ ∈ set σs
    then have sN: σ ∈ field-auto N F using ss by blast
    have sF: σ ‘ F = F by (rule field-auto-image-base[OF sN])
   — The conjugate tower is one of the family; its top is the conjugate of the original top.
    have mem: List.map ((‘) σ) Fs ∈ set Tss using s unfolding Tss-def set-map by
(rule imageI)
```

have σ ‘ *tower-top F Fs* = *tower-top F* (*List.map* ((‘) σ) *Fs*)
using *sF* **by** (*simp add*: *tower-top-conjugate*)
also have ... $\subseteq$ *tower-top F* (*spliced F Tss*)
using *spliced-top-contains-towers*[*OF sfF sfF subset-refl allt*] *mem* **by** *blast*
finally show σ ‘ *tower-top F Fs* $\subseteq$ *tower-top F* (*spliced F Tss*) **.**
qed
qed

5.5 Every element of a radical tower top is algebraic over the base

Algebraicity of a witness (Kummer/radical) chain over its base. Every element of the top *ktop F ws* of a nondegenerate witness list is algebraic over the base *F*.

theorem *ktop-algebraic-over-base*:
fixes *F* :: *complex set*
shows *Subfield F* $\Longrightarrow$ *wit-valid F ws* $\Longrightarrow$ *y* $\in$ *ktop F ws* $\Longrightarrow$ *algebraic-over F y*
proof (*induction ws arbitrary*: *F y*)
case *Nil*
then show *?case* **by** (*auto intro*: *Subfield.algebraic-over-self*)
next
case (*Cons w ws*)
then obtain *a n* **where** *an*: *w* = (*a*, *n*) **and** *n* > *0 a* ^ *n* $\in$ *F* **and** *wv*: *wit-valid* (*eval-img F a*) *ws*
by (*metis prod.collapse wit-valid.simps*(*2*))
then have *alg-aF*: *algebraic-over F a* **using** *radical-algebraic Cons* **by** *blast*
define *G* **where** *G* $\equiv$ *eval-img F a*
have *sfG*: *Subfield G* **unfolding** *G-def*
by (*simp add*: *Cons.prems*(*1*) *Subfield.subfield-eval-img alg-aF*)
— The tail runs over *G*; every element of its top is algebraic over *G* (IH).
with *Cons* **show** *?case*
using *G-def alg-aF algebraic-over-transitive an ktop-Cons wv* **by** *blast*
qed

Recast for a bare radical tower (via its witness list): every element of *tower-top F Fs* is algebraic over *F*.

theorem *radical-tower-top-algebraic*:
fixes *F* :: *complex set*
assumes *sfF*: *Subfield F* **and** *tower*: *radical-tower F Fs*
shows $\forall$ *y* $\in$ *tower-top F Fs*. *algebraic-over F y*
using *radical-tower-witnesses*[*OF sfF tower*] *ktop-algebraic-over-base ktop-eq-tower-top sfF* **by** *force*

5.6 A splitting field is normal and has a finite Galois group

theorem *splitting-field-finite-galois*:
fixes *F* :: *complex set* **and** *Phi* :: *complex poly*
assumes *sfF*: *Subfield F* **and** *Phi*: *Phi* $\in$ *poly-over F* **and** *Phinz*: *Phi* $\neq$ *0*

defines $N \equiv$ *generate-field* $(F \cup \{r.\ poly\ Phi\ r = 0\})$
shows *Subfield N F* $\subseteq N$ *finite* (*field-auto N F*)
proof −
define X **where** $X \equiv \{r.\ poly\ Phi\ r = 0\}$
have *finX*: *finite X* **unfolding** *X-def* **using** *Phinz* **by** (*rule poly-roots-finite*)
show *sfN*: *Subfield N* **unfolding** *N-def* **by** (*rule subfield-generate-field*)
show *FN*: $F \subseteq N$ **unfolding** *N-def* **using** *subset-generate-field* **by** *blast*
have *XN*: $X \subseteq N$ **unfolding** *N-def X-def* **using** *subset-generate-field* **by** *blast*
have $N =$ *gen-subfield* $(F \cup X)$
unfolding *N-def X-def* **by** (*rule gen-subfield-eq-generate-field*[*symmetric*])
then show *finite* (*field-auto N F*)
by (*metis FN Phi XN X-def finX finite-field-auto sfF sfN*)
qed

5.7 Prepending a cyclotomic step

Adjoining a root of unity is a radical step. So a radical tower *Fs* over the cyclotomic base *eval-img F* (*unity-root N*) extends, by prepending that base as a first step, to a radical tower over F itself: *eval-img F* (*unity-root N*) # *Fs*.

lemma *radical-step-cyclotomic*:
fixes F :: *complex set*
assumes *sfF*: *Subfield F* **and** *N*: $N > 0$
shows *radical-step F* (*eval-img F* (*unity-root N*))
proof −
define ζ **where** $\zeta \equiv$ *unity-root N*
have *znF*: $\zeta\ \hat{}\ N \in F$
using *N Subfield.one-closed ζ-def sfF unity-root-mem* **by** *auto*
have *alg*: *algebraic-over F ζ*
using *radical-algebraic*[*OF sfF N znF*] **.**
then show *?thesis* **unfolding** *radical-step-def ζ-def*
using *N ζ-def sfF Subfield.eval-img-eq-generate-field znF* **by** *blast*
qed

Prepending the cyclotomic step to a tower over the cyclotomic base yields a radical tower over F.

lemma *radical-tower-cyclotomic-prepend*:
fixes F :: *complex set*
assumes *sfF*: *Subfield F* **and** *N*: $N > 0$
and *tower*: *radical-tower* (*eval-img F* (*unity-root N*)) *Fs*
shows *radical-tower F* (*eval-img F* (*unity-root N*) # *Fs*)
by (*simp add*: *N radical-step-cyclotomic sfF tower*)

5.8 A radical tower embeds in a solvable normal radical top

Witness validity is monotone in the base. If *ws* is a nondegenerate witness list over F and $F \subseteq B$ with B a subfield, then *ws* is also a nondegenerate witness

list over B: the closure conditions $a^n \in F$ persist to B, and the running field *eval-img B a* still contains B (hence F).

lemma *wit-valid-exps-pos*:
fixes F :: *complex set*
shows *wit-valid F ws* $\Longrightarrow \forall x \in$ *set ws. snd x* $>$ *0*
by (*induction ws arbitrary*: *F*; *force*)

lemma *eval-img-mono-base*: $F \subseteq B \Longrightarrow$ *eval-img F a* $\subseteq$ *eval-img B a*
unfolding *eval-img-def* **using** *poly-over-mono* **by** *blast*

lemma *mono-base-lemma*:
fixes F :: *complex set*
shows *Subfield B* $\Longrightarrow F \subseteq B \Longrightarrow$ *wit-valid F ws* $\Longrightarrow$ *ktop F ws* $\subseteq$ *ktop B ws* $\wedge$ *wit-valid B ws*
proof (*induction ws arbitrary*: *F B*)
case *Nil*
then show *?case* **by** *simp*
next
case (*Cons w ws*)
then obtain *a n* **where** *an*: $w = (a, n)$ **and** $n > 0$
and $a \frown n \in B$ **and** *rest*: *wit-valid* (*eval-img F a*) *ws*
by (*metis* (*no-types*, *opaque-lifting*) *prod.exhaust subset-iff wit-valid.simps*(*2*))
then have *algebraic-over B a* **using** *radical-algebraic*
using *Cons.prems*(*1*) **by** *blast*
then have *sf*: *Subfield* (*eval-img B a*)
by (*simp add*: *Cons.prems*(*1*) *Subfield.subfield-eval-img*)
have *eval-img F a* $\subseteq$ *eval-img B a* **using** *eval-img-mono-base*[*OF Cons.prems*(*2*)] **.**
then show *?case*
using *Cons.IH Cons.prems*(*3*) ‹$a \frown n \in B$› *an sf* **by** *auto*
qed

lemma *ktop-mono-base*:
fixes F :: *complex set*
shows *Subfield B* $\Longrightarrow F \subseteq B \Longrightarrow$ *wit-valid F ws* $\Longrightarrow$ *ktop F ws* $\subseteq$ *ktop B ws*
by (*simp add*: *mono-base-lemma*)

lemma *wit-valid-mono-base*:
fixes F :: *complex set*
shows *Subfield B* $\Longrightarrow F \subseteq B \Longrightarrow$ *wit-valid F ws* $\Longrightarrow$ *wit-valid B ws*
by (*simp add*: *mono-base-lemma*)

A radical tower over a subfield F embeds into a field N that is a *solvable* radical top over F: prepend a cyclotomic step $F \rightsquigarrow F(\zeta_{N0})$ (with N_0 the product of the extracted witness exponents), realise the tower over the enlarged base by the same witnesses, and combine solvability over the cyclotomic base with solvability of the cyclotomic step via the tower step.

theorem *radical-tower-embeds-solvable-top*:
fixes F :: *complex set*
assumes *sfF*: *Subfield F* **and** *tower*: *radical-tower F Fs*

obtains N Gs **where** $Subfield\ N$ **and** $F \subseteq N$ **and** $tower\text{-}top\ F\ Fs \subseteq N$
and $radical\text{-}tower\ F\ Gs$ **and** $tower\text{-}top\ F\ Gs = N$
and $Group.solvable\ (field\text{-}auto\ N\ F)\ (compose\ N)\ (identity\ N)$
proof −
obtain ws **where** wv: $wit\text{-}valid\ F\ ws$ **and** kf: $kummer\text{-}fields\ F\ ws = Fs$
using $radical\text{-}tower\text{-}witnesses[OF\ sfF\ tower]$ **by** $blast$
define $N0$ **where** $N0 \equiv prod\text{-}list\ (List.map\ snd\ ws)$
have $N0pos$: $N0 > 0$
unfolding $N0\text{-}def$ **using** $wit\text{-}valid\text{-}exps\text{-}pos[OF\ wv]$ **by** $(induction\ ws)\ auto$
have $dvdN0$: $\forall x \in set\ ws.\ snd\ x\ dvd\ N0$
by $(simp\ add: N0\text{-}def\ prod\text{-}list\text{-}dvd)$
define ζ **where** $\zeta \equiv unity\text{-}root\ N0$
define Fz **where** $Fz \equiv eval\text{-}img\ F\ \zeta$ — The cyclotomic base
have znF: $\zeta\ \hat{}\ N0 \in F$
using $N0pos\ Subfield.one\text{-}closed\ \zeta\text{-}def\ sfF\ unity\text{-}root\text{-}mem$ **by** $auto$
have $algz$: $algebraic\text{-}over\ F\ \zeta$ **using** $radical\text{-}algebraic[OF\ sfF\ N0pos\ znF]$ **.**
have $sfFz$: $Subfield\ Fz$ **unfolding** $Fz\text{-}def$ **by** $(rule\ Subfield.subfield\text{-}eval\text{-}img[OF\ sfF\ algz])$
have FFz: $F \subseteq Fz$ **unfolding** $Fz\text{-}def$ **using** $Subfield.eval\text{-}img\text{-}base[OF\ sfF]$ **by** $auto$
— The same witnesses work over Fz; its Kummer top N is our normal radical top.
have $wvFz$: $wit\text{-}valid\ Fz\ ws$ **by** $(rule\ wit\text{-}valid\text{-}mono\text{-}base[OF\ sfFz\ FFz\ wv])$
define N **where** $N = ktop\ Fz\ ws$
have $preFz$: $pre\text{-}kummer\text{-}valid\ N0\ Fz\ ws$ **by** $(rule\ wit\text{-}valid\text{-}imp\text{-}pre\text{-}kummer\text{-}valid[OF\ wvFz\ dvdN0])$
— N is a subfield containing Fz (hence F).
have $towerFz$: $radical\text{-}tower\ Fz\ (kummer\text{-}fields\ Fz\ ws)$
by $(rule\ pre\text{-}kummer\text{-}valid\text{-}imp\text{-}radical\text{-}tower[OF\ sfFz\ preFz])$
have $Ntop\text{-}eq$: $tower\text{-}top\ Fz\ (kummer\text{-}fields\ Fz\ ws) = N$
unfolding $N\text{-}def$ **by** $(rule\ ktop\text{-}eq\text{-}tower\text{-}top[OF\ refl,\ symmetric])$
have sfN: $Subfield\ N$ **and** FzN: $Fz \subseteq N$
using $subfield\text{-}tower\text{-}top[OF\ sfFz\ towerFz]$ **by** $(auto\ simp: Ntop\text{-}eq)$
have FN: $F \subseteq N$ **using** $FFz\ FzN$ **by** $blast$
have $topFs\text{-}sub$: $tower\text{-}top\ F\ Fs \subseteq N$
by $(metis\ FFz\ N\text{-}def\ kf\ ktop\text{-}eq\text{-}tower\text{-}top\ ktop\text{-}mono\text{-}base\ sfFz\ wv)$
have zN: $unity\text{-}root\ N0 \in N$
using $FzN\ Subfield.eval\text{-}img\text{-}self[OF\ sfF]$ **by** $(auto\ simp: Fz\text{-}def\ \zeta\text{-}def)$
have $solvFzF$: $Group.solvable\ (field\text{-}auto\ Fz\ F)\ (compose\ Fz)\ (identity\ Fz)$
and $stableFz$: $\bigwedge\sigma.\ \sigma \in field\text{-}auto\ N\ F \Longrightarrow \sigma\ `\ Fz = Fz$
and $rouFz$: $\forall w.\ w\ \hat{}\ N0 = 1 \longrightarrow w \in Fz$ **unfolding** $Fz\text{-}def\ \zeta\text{-}def$
using $cyclotomic\text{-}step\text{-}solvable\text{-}and\text{-}stable[OF\ sfN\ sfF\ FN\ N0pos\ zN]$ **by** $auto$
have $solvNFz$: $Group.solvable\ (field\text{-}auto\ N\ Fz)\ (compose\ N)\ (identity\ N)$
unfolding $N\text{-}def$
by $(rule\ pre\text{-}kummer\text{-}chain\text{-}galois\text{-}solvable[OF\ sfFz\ rouFz\ preFz])$
— Combine over the tower $F \subseteq Fz \subseteq N$.
have $solvNF$: $Group.solvable\ (field\text{-}auto\ N\ F)\ (compose\ N)\ (identity\ N)$
using $FFz\ FzN\ galois\text{-}solvable\text{-}tower\text{-}step\ sfFz\ sfN\ solvFzF\ solvNFz\ stableFz$ **by** $blast$
— A radical tower over F with top N: prepend the cyclotomic step.
have $towerF$: $radical\text{-}tower\ F\ (Fz\ \#\ kummer\text{-}fields\ Fz\ ws)$
using $Fz\text{-}def\ N0pos\ \zeta\text{-}def\ radical\text{-}step\text{-}cyclotomic\ sfF\ towerFz$ **by** $force$

have *topF*: *tower-top F* (*Fz # kummer-fields Fz ws*) = *N* **using** *Ntop-eq* **by** *simp*
show *?thesis*
by (*rule that*[*OF sfN FN topFs-sub towerF topF solvNF*])
qed

5.9 Polynomial helpers: products over a subfield and their roots

lemma *poly-over-prod-list*:
assumes *Subfield K*
shows (⋀*p*. *p* ∈ *set ps* ⟹ *p* ∈ *poly-over K*) ⟹ *prod-list ps* ∈ *poly-over K*
by (*induction ps*) (*auto intro*: *Subfield.poly-over-mult*[*OF assms*] *Subfield.poly-over-1*[*OF assms*])

lemma *poly-prod-list-root-iff*:
fixes *ps* :: *'a* :: *idom poly list*
shows *poly* (*prod-list ps*) *y* = *0* ⟷ (∃ *p* ∈ *set ps*. *poly p y* = *0*)
by (*induction ps*) *auto*

lemma *prod-list-poly-nonzero*:
fixes *ps* :: *'a* :: *idom poly list*
shows (⋀*p*. *p* ∈ *set ps* ⟹ *p* ≠ *0*) ⟹ *prod-list ps* ≠ *0*
by (*induction ps*) (*auto simp*: *prod-list-zero-iff*)

Each radical base *fst w* of a witness list lies in the Kummer top.

lemma *ktop-facts*:
fixes *F* :: *complex set*
shows *Subfield F* ⟹ *wit-valid F ws* ⟹ *F* ⊆ *ktop F ws* ∧ *set* (*List.map fst ws*) ⊆ *ktop F ws*
proof (*induction ws arbitrary*: *F*)
case (*Cons w ws*)
then obtain *a n* **where** *an*: *w* = (*a*, *n*) **and** *n*: *n* > *0* **and** *a0*: *a* ≠ *0* **and** *anF*: *a* ^ *n* ∈ *F* **and** *rest*: *wit-valid* (*eval-img F a*) *ws*
by (*metis prod.exhaust wit-valid.simps*(*2*))
have *sfE*: *Subfield* (*eval-img F a*)
using *Cons.prems*(*1*) *Subfield.subfield-eval-img anF n radical-algebraic* **by** *blast*
— *a* is in the running field, hence in the top; the rest by induction.
have *aE*: *a* ∈ *eval-img F a* **by** (*rule Subfield.eval-img-self*[*OF Cons.prems*(*1*)])
then show *?case* **using** *an*
using *Cons aE rest sfE* **by** (*simp add*: *Subfield.eval-img-base subset-iff*)
qed *auto*

The base is contained in the Kummer top of a nondegenerate witness list (the chain is increasing; roots of unity are not needed for this).

lemma
assumes *Subfield F wit-valid F ws*
shows *F-subset-ktop-wit*: *F* ⊆ *ktop F ws* **and** *ktop-witness-bases-subset*: *set* (*List.map fst ws*) ⊆ *ktop F ws*
using *ktop-facts* [*OF assms*] **by** *auto*

5.10 Witness-level normal closure: bounded-exponent Kummer chain for the conjugate family

The image of a simple extension under a field homomorphism on a superfield is the generated field of the image — unconditionally (no algebraicity of the image needed).

lemma *eval-img-image-hom*:
  assumes *hom*: *field-hom-on N f* **and** *sfF*: *Subfield F* **and** *FN*: $F \subseteq N$
    and *alg*: *algebraic-over F a* **and** *aN*: $a \in N$
  shows f ‘ *eval-img F a* = *generate-field* (f ‘ $F \cup \{f\ a\}$)
proof −
  interpret *H*: *field-hom-on N f* **by** (*rule hom*)
  have *Fa-N*: $F \cup \{a\} \subseteq N$ **using** *FN aN* **by** *simp*
  have *genFa*: *eval-img F a* = *generate-field* ($F \cup \{a\}$)
    by (*rule Subfield.eval-img-eq-generate-field*[*OF sfF alg*])
  then show *?thesis*
    using *Fa-N H.image-generate-field-eq* **by** *auto*
qed

lemma *ktop-append*: *ktop F* (*xs* @ *ys*) = *ktop* (*ktop F xs*) *ys*
  by (*induction xs arbitrary*: *F*; *force*)

lemma *pre-kummer-valid-append*:
  pre-kummer-valid N0 F xs ⟹ *pre-kummer-valid N0* (*ktop F xs*) *ys*
    ⟹ *pre-kummer-valid N0 F* (*xs* @ *ys*)
  by (*induction xs arbitrary*: *F*; *force elim*: *pre-kummer-valid.elims*)

Conjugating and base-changing a witness list, at the witness level. Let σ be a field homomorphism on the ambient N fixing G setwise via σ ‘ $G = G$. Given a witness list *ws* over G whose top lies in N and whose exponents divide N_0, and a base B (a subfield of N) containing σ ‘ G, there is a witness list *cw* over B — with the *same exponents* — that is N_0-bounded (*pre-kummer-valid*) and whose Kummer top contains the conjugate σ ‘ *ktop G ws*.

lemma *base-conj-witnesses*:
  fixes N :: *complex set* **and** σ :: *complex* ⇒ *complex*
  assumes *N*: *Subfield N* **and** *hom*: *field-hom-on N σ* **and** *homN*: σ ‘ $N \subseteq N$
  shows ⋀*G B*. *Subfield G* ⟹ *Subfield B* ⟹ $G \subseteq N$ ⟹ $B \subseteq N$ ⟹ σ ‘ $G \subseteq B$
      ⟹ *wit-valid G ws* ⟹ *ktop G ws* ⊆ *N* ⟹ (∀ *x* ∈ *set ws*. *snd x dvd N0*)
      ⟹ ∃ *cw*. *pre-kummer-valid N0 B cw* ∧ σ ‘ *ktop G ws* ⊆ *ktop B cw*
          ∧ $B \subseteq$ *ktop B cw* ∧ *ktop B cw* ⊆ *N*
proof (*induction ws*)
  case (*Nil G B*)
  have σ ‘ *ktop G* [] ⊆ *ktop B* [] **using** *Nil* **by** *simp*
  moreover have $B \subseteq$ *ktop B* [] **by** *simp*
  moreover have *ktop B* [] ⊆ *N* **using** *Nil* **by** *simp*
  ultimately show *?case* **by** (*intro exI*[*of* - []]) *simp*
next
  case (*Cons w ws G B*)

interpret H: *field-hom-on N σ* **by** (*rule hom*)
obtain *a n* **where** *an*: $w = (a, n)$ **by** (*cases w*)
have *n*: $n > 0$ **and** *a0*: $a \neq 0$ **and** *anG*: $a \hat{} n \in G$ **and** *rest*: *wit-valid* (*eval-img G a*) *ws*
using *Cons.prems*(6) *an* **by** *auto*
have *ndvd*: *n dvd N0* **using** *Cons.prems*(8) *an* **by** *auto*
have *alg*: *algebraic-over G a* **using** *radical-algebraic*[*OF Cons.prems*(1) *n anG*] **.**
have *sfGa*: *Subfield* (*eval-img G a*)
by (*rule Subfield.subfield-eval-img*[*OF Cons.prems*(1) *alg*])
— a lies in N: it is in the running field, contained in the top, in N.
have *aGa*: $a \in$ *eval-img G a* **by** (*rule Subfield.eval-img-self*[*OF Cons.prems*(1)])
have *GaktopN*: *ktop* (*eval-img G a*) *ws* $\subseteq N$ **using** *Cons.prems*(7) *an* **by** *simp*
have *GaN*: *eval-img G a* $\subseteq N$
using *F-subset-ktop-wit GaktopN rest sfGa* **by** *auto*
have *aN*: $a \in N$ **using** *aGa GaN* **by** *blast*
— The conjugate radical σ a: nonzero, and its nth power lies in B.
have *sa0*: σ *a* $\neq 0$
using *H.hom-mult H.inverse-closed a0 aN* **by** *fastforce*
have *sanB*: $(\sigma\ a) \hat{} n \in B$
using *Cons.prems*(5) *H.hom-power aN anG* **by** *auto*
— The new base *eval-img B* (σ a) contains σ ‘ *eval-img G a*.
define B' **where** $B' \equiv$ *eval-img B* (σ a)
have *algsa*: *algebraic-over B* (σ a) **using** *radical-algebraic*[*OF Cons.prems*(2) *n sanB*] **.**
have *sfB'*: *Subfield* B' **unfolding** B'*-def* **by** (*rule Subfield.subfield-eval-img*[*OF Cons.prems*(2) *algsa*])
have *BB'*: $B \subseteq B'$ **unfolding** B'*-def* **using** *Subfield.eval-img-base*[*OF Cons.prems*(2)] **by** *auto*
have *saB'*: $\sigma\ a \in B'$ **unfolding** B'*-def* **by** (*rule Subfield.eval-img-self*[*OF Cons.prems*(2)])
— The conjugate simple extension is generated by the images and lands in the new base B'.
have *img-eq*: σ ‘ *eval-img G a* = *generate-field* (σ ‘ $G \cup \{\sigma\ a\}$)
by (*rule eval-img-image-hom*[*OF hom Cons.prems*(1) *Cons.prems*(3) *alg aN*])
have B'*gen*: B' = *generate-field* ($B \cup \{\sigma\ a\}$)
unfolding B'*-def* **by** (*rule Subfield.eval-img-eq-generate-field*[*OF Cons.prems*(2) *algsa*])
have *sGaB'*: σ ‘ *eval-img G a* $\subseteq B'$
proof −
have σ ‘ $G \cup \{\sigma\ a\} \subseteq B \cup \{\sigma\ a\}$ **using** *Cons.prems*(5) **by** *blast*
then show *?thesis* **by** (*metis img-eq B'gen generate-field-mono*)
qed
have $B'N$: $B' \subseteq N$
proof −
have $\sigma\ a \in N$ **using** *aN homN* **by** *blast*
then show *?thesis* **using** *Cons.prems*(4) B'*gen generate-field-least*[*OF N*] **by** *simp*
qed
— Recurse: witnesses over B' for the tail, conjugating *eval-img G a*.
have *tail-bound*: $\forall x \in$ *set ws*. *snd x dvd N0* **using** *Cons.prems*(8) **by** *auto*
obtain *cw* **where** *cw*: *pre-kummer-valid N0 B' cw*
σ ‘ *ktop* (*eval-img G a*) *ws* $\subseteq$ *ktop B' cw B'* $\subseteq$ *ktop B' cw ktop B' cw* $\subseteq N$

```
  using Cons.IH[OF sfGa sfB' GaN B'N sGaB' rest GaktopN tail-bound] by blast
 — Assemble: prepend the step (σ a, n).
 have pre: pre-kummer-valid N0 B ((σ a, n) # cw)
  using n ndvd sa0 sanB cw(1) by (simp add: B'-def)
 have top: σ ‘ ktop G (w # ws) ⊆ ktop B ((σ a, n) # cw)
  using an cw(2) by (simp add: B'-def)
 have base: B ⊆ ktop B ((σ a, n) # cw) using BB' cw(3) by (simp add: B'-def)
 have topN: ktop B ((σ a, n) # cw) ⊆ N using cw(4) by (simp add: B'-def)
 show ?case using pre top base topN by blast
qed
```

Folding the conjugate witnesses over the whole Galois group. Given a base tower *ws* over *G*, a cyclotomic base *Fz* (a subfield of *N* containing σ ‘ *G* for every σ in the list), we accumulate the conjugate witness lists of σ ‘ *ktop G ws* for all σ into one N_0-bounded pre-Kummer chain over *Fz* whose top stays inside *N* and contains every conjugate σ ‘ *ktop G ws*. Induction on the list, appending each *base-conj-witnesses* chain with *pre-kummer-valid-append* / *ktop-append*.

```
lemma conj-family-witnesses:
 fixes N Fz G :: complex set and ws :: (complex × nat) list
 assumes N: Subfield N and sfG: Subfield G and GN: G ⊆ N
  and wv: wit-valid G ws and topGN: ktop G ws ⊆ N and dvd: ∀x∈set ws. snd x dvd N0
 shows ⋀Fz. Subfield Fz ⟹ Fz ⊆ N ⟹ G ⊆ Fz ⟹ (∀σ∈set σs. σ ∈ field-auto N G)
        ⟹ ∃CW. pre-kummer-valid N0 Fz CW ∧ Fz ⊆ ktop Fz CW ∧ ktop Fz CW ⊆ N
                 ∧ (∀σ∈set σs. σ ‘ ktop G ws ⊆ ktop Fz CW)
proof (induction σs)
 case (Nil Fz)
 show ?case by (intro exI[of - []]) (use Nil in simp)
next
 case (Cons σ σs Fz)
 obtain CW0 where CW0: pre-kummer-valid N0 Fz CW0 Fz ⊆ ktop Fz CW0 ktop Fz CW0 ⊆ N
     ∀τ∈set σs. τ ‘ ktop G ws ⊆ ktop Fz CW0
   using Cons.IH[OF Cons.prems(1,2,3)] Cons.prems(4) by auto
 define B where B ≡ ktop Fz CW0
 have sfB: Subfield B
 proof −
  have radical-tower Fz (kummer-fields Fz CW0)
   by (rule pre-kummer-valid-imp-radical-tower[OF Cons.prems(1) CW0(1)])
  then show ?thesis
    using subfield-tower-top[OF Cons.prems(1)] by (force simp: B-def ktop-def tower-top-def)
 qed
 — Conjugate ktop G ws into a over B.
 have hom: field-hom-on N σ using N Cons using field-auto-imp-field-hom-on by fastforce
 have sNN: σ ‘ N ⊆ N
```

using *field-auto-closed Cons* **by** *auto*
have *sGB*: σ ‘ $G \subseteq B$
using *B-def CW0(2) Cons.prems field-auto-image-base* **by** *fastforce*
have *BN*: $B \subseteq N$ **unfolding** *B-def* **by** (*rule CW0(3)*)
obtain *cw* **where** *cw*: *pre-kummer-valid N0 B cw* σ ‘ *ktop G ws* $\subseteq$ *ktop B cw*
$B \subseteq$ *ktop B cw ktop B cw* $\subseteq N$
using *base-conj-witnesses*[*OF N hom sNN sfG sfB GN BN sGB wv topGN dvd*]
by *blast*
— Append the two chains.
define *CW* **where** *CW* = *CW0* @ *cw*
have *pre*: *pre-kummer-valid N0 Fz CW* **unfolding** *CW-def B-def*
using *B-def CW0(1) cw(1) pre-kummer-valid-append* **by** *blast*
then have *covers*: *Fz* $\subseteq$ *ktop Fz CW* $\forall \tau \in$ *set* (σ # σs). τ ‘ *ktop G ws* $\subseteq$ *ktop Fz CW ktop Fz CW* $\subseteq N$
using *B-def CW-def ktop-append CW0 cw* **by** *force+*
with *pre* **show** *?case* **by** *blast*
qed

The Kummer top of a pre-Kummer chain is a subfield (each step is a simple algebraic extension; roots of unity are not needed).

lemma *subfield-ktop-pre*:
fixes *F* :: *complex set*
assumes *Subfield F* **and** *pre-kummer-valid N0 F ws*
shows *Subfield* (*ktop F ws*)
proof −
have *Subfield* (*tower-top F* (*kummer-fields F ws*))
using *assms pre-kummer-valid-imp-radical-tower subfield-tower-top* **by** *blast*
then show *?thesis* **by** (*simp add*: *ktop-def tower-top-def*)
qed

If the base lies in a subfield N and every radical base of the witness list lies in N, then the whole Kummer top lies in N: each simple step *eval-img G a* = *generate-field* ($G \cup \{a\}$) stays inside N.

lemma *ktop-subset-of-bases*:
fixes *N* :: *complex set*
assumes *N*: *Subfield N*
shows $\bigwedge$*F*. *Subfield F* $\Longrightarrow$ *F* $\subseteq$ *N* $\Longrightarrow$ *wit-valid F ws* $\Longrightarrow$ *set* (*List.map fst ws*) $\subseteq$ *N*
$\Longrightarrow$ *ktop F ws* $\subseteq N$
proof (*induction ws*)
case *Nil*
then show *?case* **by** *simp*
next
case (*Cons w ws F*)
then obtain *a n* **where** *an*: $w = (a, n)$ **and** *n*: $n > 0$ **and** *anF*: $a \frown n \in F$
and *rest*: *wit-valid* (*eval-img F a*) *ws*
by (*metis surj-pair wit-valid.simps*(*2*))
have *aN*: $a \in N$ **using** *Cons.prems*(*4*) *an* **by** *simp*
have *alg*: *algebraic-over F a* **using** *radical-algebraic*[*OF Cons.prems*(*1*) *n anF*] **.**

have *sfEa*: *Subfield* (*eval-img F a*) **by** (*rule Subfield.subfield-eval-img*[*OF Cons.prems*(*1*) *alg*])
— The one-step extension *eval-img F a* lands in *N*.
have *eval-img F a* = *generate-field* (*F* ∪ {*a*})
by (*rule Subfield.eval-img-eq-generate-field*[*OF Cons.prems*(*1*) *alg*])
moreover have *F* ∪ {*a*} ⊆ *N* **using** *Cons.prems*(*2*) *aN* **by** *simp*
ultimately have *EaN*: *eval-img F a* ⊆ *N* **using** *generate-field-least*[*OF N*] **by** *simp*
with *Cons* **show** *?case* **using** *an sfEa* **by** *auto*
qed

5.11 The normal radical closure of a radical tower is a solvable splitting field

Suppose *Fs* is a radical tower over *F* whose top contains the finite root set *X* of an irreducible *F*-polynomial *P*. Let *Kq* = *generate-field* (*F* ∪ *X*). Then there is a normal closure *N* and an *F*-polynomial Ψ exhibiting *N* as the splitting field *generate-field* (*Kq* ∪ {*r*. *poly* Ψ *r* = *0*}) over *Kq*, with *N* solvable over *F*.

theorem *radical-tower-normal-closure-solvable*:
fixes *F* :: *complex set* **and** *P* :: *complex poly*
assumes *sfF*: *Subfield F* **and** *tower*: *radical-tower F Fs*
and *P*: *P* ∈ *poly-over F* **and** *irrP*: *irreducible-over F P*
and *Xdef*: *X* = {*r*. *poly P r* = *0*} **and** *finX*: *finite X* **and** *XKq*: *X* ⊆ *tower-top F Fs*
and *closed*: ⋀*q* :: *complex poly*. *0* < *degree q* ⟹ ∃ *b*. *poly q b* = *0*
defines *Kq* ≡ *generate-field* (*F* ∪ *X*)
obtains *N* Ψ **where** *Subfield N* **and** *Kq* ⊆ *N*
and Ψ ∈ *poly-over F*
and *N* = *generate-field* (*Kq* ∪ {*r*. *poly* Ψ *r* = *0*})
and *finite* {*r*. *poly* Ψ *r* = *0*}
and ⋀*r*. *poly* Ψ *r* = *0* ⟹ *algebraic-over Kq r*
and *Group.solvable* (*field-auto N F*) (*compose N*) (*identity N*)
proof −
interpret *F*: *Subfield F* **by** (*rule sfF*)
— The radical generators, extracted as the bases of a witness list; each is algebraic over *F*.
obtain *ws* **where** *wv*: *wit-valid F ws* **and** *kf*: *kummer-fields F ws* = *Fs*
using *radical-tower-witnesses*[*OF sfF tower*] **by** *blast*
define *as* **where** *as* ≡ *List.map fst ws*
have *ktop-eq*: *ktop F ws* = *tower-top F Fs* **by** (*rule ktop-eq-tower-top*[*OF kf*])
— Each radical base lies in the tower top.
have *as-sub*: *set as* ⊆ *tower-top F Fs*
using *as-def ktop-eq ktop-witness-bases-subset sfF wv* **by** *blast*
have *alg-top*: ∀ *y* ∈ *tower-top F Fs*. *algebraic-over F y*
by (*rule radical-tower-top-algebraic*[*OF sfF tower*])
have *algas*: ⋀*a*. *a* ∈ *set as* ⟹ *algebraic-over F a* **using** *as-sub alg-top* **by** *blast*
— The witnessing polynomial: the given irreducible *P*, the cyclotomic factor for the exponent bound, and the minimal polynomials of the radical generators.
define *N0* **where** *N0* = *prod-list* (*List.map snd ws*)
have *N0pos*: *N0* > *0*

unfolding *N0-def* **using** *wit-valid-exps-pos*[*OF wv*] **by** (*induction ws*) *auto*
define *cyc* **where** *cyc* ≡ (*monom 1 N0 − 1* :: *complex poly*)
define Ψ **where** Ψ ≡ *P* ∗ *cyc* ∗ *prod-list* (*List.map* (*minpoly F*) *as*)
— Ψ is a nonzero polynomial over *F*.
have *cycF*: *cyc* ∈ *poly-over F*
unfolding *cyc-def* **using** *F.poly-over-monom*[*of 1 N0*] *F.one-closed F.poly-over-diff F.poly-over-1*
by *simp*
have *minF*: ⋀*a*. *a* ∈ *set as* ⟹ *minpoly F a* ∈ *poly-over F*
by (*rule Subfield.minpoly-over*[*OF sfF algas*])
have *prodF*: *prod-list* (*List.map* (*minpoly F*) *as*) ∈ *poly-over F*
by (*rule poly-over-prod-list*[*OF sfF*]) (*auto simp*: *minF*)
have *PsiF*: Ψ ∈ *poly-over F*
unfolding Ψ*-def* **using** *P cycF prodF* **by** (*auto intro*: *F.poly-over-mult*)
have *cyc-eval*: *poly cyc 0* = − *1*
unfolding *cyc-def* **using** *N0pos* **by** (*simp add*: *poly-monom*)
have *cyc-nz*: *cyc* ≠ *0* **using** *cyc-eval* **by** (*metis poly-0 neg-equal-0-iff-equal one-neq-zero*)
have *P-nz*: *P* ≠ *0* **using** *irrP* **by** (*simp add*: *irreducible-over-def*)
have *minpoly-nz*: ⋀*a*. *a* ∈ *set as* ⟹ *minpoly F a* ≠ *0*
by (*rule Subfield.minpoly-nonzero*[*OF sfF algas*])
have *prod-nz*: *prod-list* (*List.map* (*minpoly F*) *as*) ≠ *0*
by (*metis imageE list.set-map minpoly-nz prod-list-poly-nonzero*)
have *Psi-nz*: Ψ ≠ *0* **unfolding** Ψ*-def* **using** *P-nz cyc-nz prod-nz* **by** *simp*
— Its root set.
define *R* **where** *R* = {*r*. *poly* Ψ *r* = *0*}
have *finR*: *finite R* **unfolding** *R-def* **using** *Psi-nz* **by** (*rule poly-roots-finite*)
— *X* (roots of *P*) is among the roots of Ψ.
have *XR*: *X* ⊆ *R*
using *R-def Xdef* Ψ*-def* **by** *force*
— The splitting field.
define *N* **where** *N* ≡ *generate-field* (*F* ∪ *R*)
have *sfN*: *Subfield N* **unfolding** *N-def* **by** (*rule subfield-generate-field*)
have *R* ⊆ *N* **unfolding** *N-def* **using** *subset-generate-field* **by** *blast*
— *Kq* lies in *N* and *N* is the splitting field of Ψ over *Kq*.
have *KqN*: *Kq* ⊆ *N*
unfolding *Kq-def N-def* **using** *XR generate-field-mono*[*of F* ∪ *X F* ∪ *R*] **by** *blast*
have *N-split-Kq*: *N* ≡ *generate-field* (*Kq* ∪ *R*)
by (*simp add*: *Kq-def N-def XR sup.assoc sup.absorb2*)
— Roots of Ψ are algebraic over *F*, hence over *Kq*.
have *FKq*: *F* ⊆ *Kq* **unfolding** *Kq-def* **using** *subset-generate-field* **by** *blast*
have *algR-Kq*: ⋀*r*. *poly* Ψ *r* = *0* ⟹ *algebraic-over Kq r*
by (*meson FKq PsiF Psi-nz algebraic-over-def algebraic-over-mono*)
— Key objective: solvability of the Galois group of the splitting field *N* over *F*.
have *FN*: *F* ⊆ *N* **unfolding** *N-def* **using** *subset-generate-field* **by** *blast*
have *dvdN0*: ∀ *x* ∈ *set ws*. *snd x dvd N0*
by (*simp add*: *N0-def prod-list-dvd*)
define ζ **where** ζ ≡ *unity-root N0*
define *Fz* **where** *Fz* ≡ *eval-img F* ζ
— Step 0: the cyclotomic base *Fz* sits inside *N* (the roots of unity are roots of the cyclotomic factor *cyc* of Ψ).

have *zroot*: $\zeta \mathbin{\hat{}} N0 = 1$ **unfolding** *ζ-def* **using** *unity-root-mem*[*OF N0pos*] **by** *simp*
have *algz*: *algebraic-over F ζ*
using *radical-algebraic*[*OF sfF N0pos*] **by** (*simp add*: *zroot*)
have *sfFz*: *Subfield Fz* **unfolding** *Fz-def* **by** (*rule Subfield.subfield-eval-img*[*OF sfF algz*])
have *FFz*: $F \subseteq Fz$ **unfolding** *Fz-def* **using** *Subfield.eval-img-base*[*OF sfF*] **by** *auto*
have *zetaR*: $\zeta \in R$
using *zroot* **by** (*simp add*: *cyc-def R-def Ψ-def poly-monom*)
have *zetaN*: $\zeta \in N$ **using** *zetaR* ‹$R \subseteq N$› **by** *blast*
have *Fz-gen*: $Fz = generate\text{-}field\ (F \cup \{\zeta\})$
unfolding *Fz-def* **by** (*rule Subfield.eval-img-eq-generate-field*[*OF sfF algz*])
have *FzN*: $Fz \subseteq N$
by (*simp add*: *FN Fz-gen generate-field-least sfN zetaN*)
— Step 1: N is the splitting field of Ψ over Fz, so *field-auto N Fz* is finite; likewise over F.
have *PsiFz*: $\Psi \in poly\text{-}over\ Fz$ **using** *PsiF poly-over-mono*[*OF FFz*] **by** *blast*
have *N-split-Fz*: $N = generate\text{-}field\ (Fz \cup R)$
by (*simp add*: *Fz-gen N-def insert-absorb zetaR*)
have *finGalF*: *finite* (*field-auto N F*)
using *splitting-field-finite-galois*[*OF sfF PsiF Psi-nz*] **unfolding** *N-def R-def* **by** *simp*
— Step 2: enumerate the (finite) Galois group *field-auto N F*.
obtain *σs* **where** *σs*: *set σs* = *field-auto N F* **using** *finite-list*[*OF finGalF*] **by** *blast*
— Step 3: each radical base is a root of its minimal polynomial (a factor of Ψ), hence in $R \subseteq N$; so the base tower *ws* lies in N.
have *asR*: *set as* $\subseteq R$
using *F.minpoly-root algas poly-prod-list-root-iff*
by (*fastforce simp*: *Ψ-def R-def*)
have *asN*: *set as* $\subseteq N$ **using** *asR* ‹$R \subseteq N$› **by** *blast*
have *wsN*: *ktop F ws* $\subseteq N$
using *FN asN as-def ktop-subset-of-bases sfF sfN wv* **by** *auto*
— Step 4: fold the conjugate witnesses over the whole group, giving a pre-Kummer chain *CW* over *Fz* whose top *ktop Fz CW* lies in N and covers every conjugate.
have *σsF*: $\forall \sigma \in set\ \sigma s.\ \sigma \in field\text{-}auto\ N\ F$ **using** *σs* **by** *simp*
obtain *CW* **where** *CW*: *pre-kummer-valid N0 Fz CW Fz* $\subseteq$ *ktop Fz CW ktop Fz CW* $\subseteq N$
$\forall \sigma \in set\ \sigma s.\ \sigma$ ‘ *ktop F ws* $\subseteq$ *ktop Fz CW*
using *conj-family-witnesses*[*OF sfN sfF FN wv wsN dvdN0 sfFz FzN FFz σsF*] **by** *blast*
define *T* **where** $T \equiv ktop\ Fz\ CW$
have *sfT*: *Subfield T* **unfolding** *T-def* **by** (*rule subfield-ktop-pre*[*OF sfFz CW*(*1*)])
— The cyclotomic step over F: solvable, stable, and supplying the roots of unity.
have *zetaN′*: *unity-root N0* $\in N$ **using** *zetaN* **by** (*simp add*: *ζ-def*)
have *Fz-eq*: $Fz = eval\text{-}img\ F\ (unity\text{-}root\ N0)$ **by** (*simp add*: *Fz-def ζ-def*)
have *rouFz*: $\forall w.\ w \mathbin{\hat{}} N0 = 1 \longrightarrow w \in Fz$
using *cyclotomic-step-solvable-and-stable*[*OF sfN sfF FN N0pos zetaN′*]
unfolding *Fz-eq* **by** *force*
— Step 5: $N = T$.
have *FzT*: $Fz \subseteq T$ **unfolding** *T-def* **by** (*rule CW*(*2*))
— The identity automorphism belongs to the enumerated group; the base tower top

lies in T.
have *wsT*: *ktop F ws* $\subseteq$ *T*
proof −
have *cN*: *complex-subfield N* **using** *sfN* **by** (*simp add*: *complex-subfield-iff-subfield*)
have *idg*: *identity N* $\in$ *set* σs **using** *identity-field-auto*[*OF cN FN*] σs **by** *simp*
have *img*: *identity N* ‘ *ktop F ws* $\subseteq$ *ktop Fz CW* **using** *CW*(*4*) *idg* **by** *blast*
show *?thesis*
using *T-def img wsN* **by** *auto*
qed
— *Conjugacy.* Each root b of *minpoly F a* (for a radical generator a) is a Galois conjugate $\sigma\ a$ of a under some $\sigma \in$ *field-auto N F*, hence lies in σ ‘ *ktop F ws* $\subseteq T$ (as $a \in$ *ktop F ws*).
have *minroot-T*: *b* $\in$ *T* **if** *a*: *a* $\in$ *set as* **and** *mb*: *poly* (*minpoly F a*) *b* = *0* **for** *a b*
proof −
define *m* **where** *m* $\equiv$ *minpoly F a*
have *alga*: *algebraic-over F a* **by** (*rule algas*[*OF a*])
have *irrm*: *irreducible-over F m* **unfolding** *m-def* **by** (*rule irreducible-over-minpoly*[*OF sfF alga*])
have *mF*: *m* $\in$ *poly-over F* **unfolding** *m-def* **by** (*rule Subfield.minpoly-over*[*OF sfF alga*])
have *ma*: *poly m a* = *0* **unfolding** *m-def* **by** (*rule Subfield.minpoly-root*[*OF sfF alga*])
define *Rm* **where** *Rm* $\equiv$ {*r*. *poly m r* = *0*}
have *mnz*: *m* $\neq$ *0* **unfolding** *m-def* **by** (*rule Subfield.minpoly-nonzero*[*OF sfF alga*])
have *finRm*: *finite Rm* **unfolding** *Rm-def* **using** *mnz* **by** (*rule poly-roots-finite*)
have *algRm*: $\bigwedge$*r*. *r* $\in$ *Rm* $\Longrightarrow$ *algebraic-over F r*
unfolding *Rm-def* **using** *mF mnz* **unfolding** *algebraic-over-def* **by** *blast*
have *aRm*: *a* $\in$ *Rm* **using** *ma* **by** (*simp add*: *Rm-def*)
have *bRm*: *b* $\in$ *Rm* **using** *mb* **by** (*simp add*: *Rm-def m-def*)
— The roots of m all lie in N (m divides Ψ).
have *RmR*: *Rm* $\subseteq$ *R*
proof
fix *r* **assume** *r* $\in$ *Rm*
then have *poly m r* = *0* **by** (*simp add*: *Rm-def*)
then have *poly* (*prod-list* (*List.map* (*minpoly F*) *as*)) *r* = *0*
using *a* **by** (*force simp*: *m-def poly-prod-list-root-iff*)
then show *r* $\in$ *R* **by** (*simp add*: Ψ*-def R-def*)
qed
have *RmN*: *Rm* $\subseteq$ *N* **using** *RmR* ‹*R* $\subseteq$ *N*› **by** *blast*
— Transitivity: an F-homomorphism on the root-generated field sending a to b.
have *Rm-eq*: *Rm* = {*r*. *poly m r* = *0*} **by** (*simp add*: *Rm-def*)
obtain σ **where** *hom0*: *field-hom-on* (*generate-field* (*F* $\cup$ *Rm*)) σ
and *s-ab*: $\sigma\ a = b$ **and** *fixF*: $\forall x{\in}F.\ \sigma\ x = x$
and *onto0*: σ ‘ *generate-field* (*F* $\cup$ *Rm*) = *generate-field* (*F* $\cup$ *Rm*)
using *galois-transitive-irreducible*[*OF sfF finRm algRm mF irrm Rm-eq closed aRm bRm*]
by *blast*
— Extend σ over the remaining roots of Ψ to all of N.
define *K0* **where** *K0* $\equiv$ *generate-field* (*F* $\cup$ *Rm*)

have *algRmore*: *algebraic-over K0 r* **if** $r \in R$ **for** r
proof −
have *algebraic-over F r* **using** *that PsiF Psi-nz R-def* **unfolding** *algebraic-over-def* **by** *blast*
then show *algebraic-over K0 r*
by (*metis K0-def algebraic-over-mono le-sup-iff subset-generate-field*)
qed
obtain *g* **where** *homg*: *field-hom-on* (*generate-field* (*K0* ∪ *R*)) *g* **and** *gK0*: $\forall x{\in}K0.\ g\ x = \sigma\ x$
using *field-hom-on-extend-finite*[*OF closed finR hom0*[*folded K0-def*] *algRmore*] **by** *blast*
— *generate-field* (*K0* ∪ *R*) = *N*.
have *K0R-N*: *generate-field* (*K0* ∪ *R*) = *N*
by (*simp add*: *K0-def N-def RmR Un-assoc sup.absorb2*)
have *homgN*: *field-hom-on N g* **using** *homg* **by** (*simp add*: *K0R-N*)
have *fixFg*: $\bigwedge x.\ x \in F \Longrightarrow g\ x = x$
by (*simp add*: *K0-def fixF gK0 generate-field.generate-field-base*)
— *g* maps *R* into *R* (it fixes *F* and preserves roots of Ψ), and injectively, so onto *R*; hence *g* maps *N* onto *N*.
have *gR-sub*: *g* ‘ *R* ⊆ *R*
using *hom-preserves-roots*[*OF homgN sfF FN fixFg PsiF*] ‹*R* ⊆ *N*›
by (*auto simp*: *R-def*)
have *injR*: *inj-on g R*
using ‹*R* ⊆ *N*› *field-hom-on.inj-on homgN inj-on-subset* **by** *blast*
have *gR*: *g* ‘ *R* = *R* **by** (*rule endo-inj-surj*[*OF finR gR-sub injR*])
have *gUn*: *g* ‘ (*F* ∪ *R*) = *F* ∪ *R* **using** *gR* **by** (*simp add*: *fixFg image-Un*)
have *FR-N*: *F* ∪ *R* ⊆ *N* **using** *FN* ‹*R* ⊆ *N*› **by** *simp*
have *gN*: *g* ‘ *N* = *N*
by (*metis FR-N N-def field-hom-on.image-generate-field-eq gUn homgN*)
— So *restrict g N* is an *F*-automorphism of *N* sending *a* to *b*.
have *aut*: *restrict g N* ∈ *field-auto N F*
by (*rule field-hom-on-imp-field-auto*[*OF homgN gN fixFg FN*])
have *a* ∈ *N* **using** *a asN* **by** *blast*
have *aws*: *a* ∈ *ktop F ws*
using *ktop-witness-bases-subset*[*OF sfF wv*] *a* **by** (*auto simp*: *as-def*)
have *restrict g N a* = *b*
using *K0-def* ‹*a* ∈ *N*› *aRm gK0 s-ab* **by** *auto*
then show *?thesis*
— *b* is the image of *a* ∈ *ktop F ws* under an automorphism in the family.
using *CW*(*4*) *T-def σs aut aws* **by** *blast*
qed
— Step 5: *N* = *T*.
have *RT*: *R* ⊆ *T*
proof
fix *r* **assume** *r* ∈ *R*
then consider *poly P r = 0* | *poly cyc r = 0* | *poly* (*prod-list* (*List.map* (*minpoly F*) *as*)) *r = 0*
by (*force simp*: Ψ*-def R-def*)
then show *r* ∈ *T*
proof *cases*

```
    case 1
    then show ?thesis
      using XKq Xdef ktop-eq wsT by auto
  next
    case 2
    then have r ^ N0 = 1 unfolding cyc-def by (simp add: poly-monom)
    then show ?thesis using rouFz FzT by blast
  next
    case 3
    then show ?thesis
      using minroot-T by (auto simp: poly-prod-list-root-iff)
  qed
 qed
 have F ⊆ T using FFz FzT by blast
 then have F ∪ R ⊆ T using RT by blast
 then have NT: N=T
    using generate-field-least sfT CW(3) unfolding N-def T-def by (metis subset-antisym)
  — Step 6: solvability of N over Fz (roots of unity in Fz).
 have solvNFz: Group.solvable (field-auto N Fz) (compose N) (identity N)
  using pre-kummer-chain-galois-solvable[OF sfFz rouFz CW(1)] NT by (simp flip: T-def)
 — Step 7: combine with the solvable cyclotomic step over F.
 obtain solvFzF: Group.solvable (field-auto Fz F) (compose Fz) (identity Fz)
   and stableFz: ⋀σ. σ ∈ field-auto N F ⟹ σ ‘ Fz = Fz
   using Fz-eq N0pos NT cyclotomic-step-solvable-and-stable sfF
   using ‹F ⊆ T› sfT zetaN' by presburger
 have solvN: Group.solvable (field-auto N F) (compose N) (identity N)
   using galois-solvable-tower-step[where K = N and E = Fz and F = F]
   using FFz FzN sfFz sfN solvFzF solvNFz stableFz by blast
 show ?thesis
   using N-split-Kq PsiF R-def algR-Kq finR solvN that by auto
qed
```

The preceding construction is the reusable bridge from a radical tower to the Galois group of the field generated by the roots of an irreducible polynomial. The normal radical top is an auxiliary object: after its solvability is established, the splitting-field restriction theorem descends solvability to the root field itself.

```
theorem radical-tower-root-field-solvable:
 fixes F :: complex set and P :: complex poly
 assumes sfF: Subfield F and tower: radical-tower F Fs
   and P: P ∈ poly-over F and irrP: irreducible-over F P
   and Xdef: X = {r. poly P r = 0} and finX: finite X
   and Xtop: X ⊆ tower-top F Fs
   and closed: ⋀q :: complex poly. 0 < degree q ⟹ ∃ b. poly q b = 0
 defines Kq ≡ generate-field (F ∪ X)
 shows Group.solvable (field-auto Kq F) (compose Kq) (identity Kq)
proof −
 obtain N Psi where sfN: Subfield N and KqN: Kq ⊆ N
```

```
    and Psi: Psi ∈ poly-over F
    and N: N = generate-field (Kq ∪ {r. poly Psi r = 0})
    and finR: finite {r. poly Psi r = 0}
    and solvN: Group.solvable (field-auto N F) (compose N) (identity N)
  unfolding Kq-def
  by (blast intro: radical-tower-normal-closure-solvable
        [OF sfF tower P irrP Xdef finX Xtop closed, unfolded Kq-def])
 have sfKq: Subfield Kq
  unfolding Kq-def by (rule subfield-generate-field)
 have FKq: F ⊆ Kq
  unfolding Kq-def using subset-generate-field by blast
 have Psi-nz: Psi ≠ 0
  using finR infinite-UNIV-char-0 by fastforce
 have Psi ∈ poly-over Kq
  using Psi poly-over-mono[OF FKq] by blast
 then have splitN: splitting-field Kq Psi N
  unfolding splitting-field-def
  using Psi-nz N by (simp add: gen-subfield-eq-generate-field poly-root-set-def)
 have X ⊆ Kq
  unfolding Kq-def using subset-generate-field by blast
 then have XN: X ⊆ N using KqN by blast
 have stableKq: ⋀σ. σ ∈ field-auto N F ⟹ σ ` Kq = Kq
   using root-generated-stable[OF sfN sfF - P Xdef XN finX] FKq KqN by (auto
simp: Kq-def)
 show ?thesis
   by (rule galois-splitting-subfield-solvable[OF sfKq sfF FKq Psi splitN stableKq
solvN])
qed

end
```

6 A Concrete Quintic with Galois Group S_5

```
theory Quintic-Example
 imports
   HOL−New-Algebra.Complex-Field-Extension
   HOL−New-Algebra.Abel-Ruffini
   HOL−New-Algebra.Galois-Transitivity
   HOL−Computational-Algebra.Polynomial-Factorial
   Sturm-Sequences.Sturm
begin
```

We study the quintic $x^5 - 4x + 2$ over the rationals. It is irreducible (Eisenstein at *2*, proved here directly) and has exactly three real roots and one pair of complex-conjugate roots (the real-root count via the *sturm* method of *Sturm-Sequences.Sturm*). These are the hypotheses from which its Galois group is shown to be S_5.

6.1 Eisenstein's irreducibility criterion

A general form of Eisenstein's criterion over the integers: a polynomial satisfying the Eisenstein conditions for a prime p has no factorisation into two factors of positive degree.

lemma *sum-atMost-split*: $(\sum l{\leq}(i::nat).\ F\ l) = F\ i + (\sum l{<}i.\ F\ l)$
by (*metis add.commute lessThan-Suc-atMost sum.lessThan-Suc*)

lemma *eisenstein-factor-helper*:
fixes *f g h* :: *int poly* **and** *p* :: *int*
assumes *prime*: *prime p*
and *fgh*: $f = g * h$
and *lead*: ¬ *p dvd lead-coeff g*
and *mid*: $\bigwedge i.\ i < degree\ f \Longrightarrow p\ dvd\ coeff\ f\ i$
and *ph0*: ¬ *p dvd coeff h 0*
and *dg*: $0 < degree\ g$ **and** *dh*: $0 < degree\ h$
shows *False*
proof −
have *g0*: $g \neq 0$ **and** *h0*: $h \neq 0$ **using** *dg dh* **by** *auto*
have *df*: *degree f* = *degree g* + *degree h* **using** *fgh g0 h0* **by** (*simp add*: *degree-mult-eq*)
have *all-dvd*: *p dvd coeff g i* **if** $i \leq degree\ g$ **for** *i*
using *that*
proof (*induct i rule*: *less-induct*)
case (*less i*)
have *ilt*: $i < degree\ f$ **using** *less.prems df dh* **by** *simp*
have *coeff f i* = $(\sum l{\leq}i.\ coeff\ g\ l * coeff\ h\ (i - l))$ **using** *fgh* **by** (*simp add*: *coeff-mult*)
also have ... = *coeff g i* * *coeff h 0* + $(\sum l{<}i.\ coeff\ g\ l * coeff\ h\ (i - l))$
by (*subst sum-atMost-split*) *simp*
finally have *eq*: *coeff f i* = *coeff g i* * *coeff h 0* + $(\sum l{<}i.\ coeff\ g\ l * coeff\ h\ (i - l))$ **.**
have *psum*: *p dvd* $(\sum l{<}i.\ coeff\ g\ l * coeff\ h\ (i - l))$
by (*intro dvd-sum*) (*use less.hyps less.prems* **in** *auto*)
then have *p dvd coeff g i* * *coeff h 0*
by (*metis* (*lifting*) *eq psum ilt mid add.commute dvd-add-right-iff*)
then show *p dvd coeff g i* **using** *ph0 prime* **by** (*auto dest*: *prime-dvd-multD*)
qed
have *p dvd lead-coeff g* **using** *all-dvd*[*of degree g*] **by** *simp*
with *lead* **show** *False* **..**
qed

lemma *eisenstein-no-factor*:
fixes *f g h* :: *int poly* **and** *p* :: *int*
assumes *prime*: *prime p*
and *fgh*: $f = g * h$
and *lead*: ¬ *p dvd lead-coeff f*
and *mid*: $\bigwedge i.\ i < degree\ f \Longrightarrow p\ dvd\ coeff\ f\ i$
and *const*: ¬ *p*^2 *dvd coeff f 0*

```
    and dg: 0 < degree g and dh: 0 < degree h
  shows False
proof −
  have lead-coeff f = lead-coeff g * lead-coeff h using fgh by (simp add: lead-coeff-mult)
  then have lg: ¬ p dvd lead-coeff g and lh: ¬ p dvd lead-coeff h
    using lead by (auto simp: prime-dvd-mult-iff[OF prime])
  have coeff f 0 = coeff g 0 * coeff h 0 using fgh by (simp add: coeff-mult-0)
  then have ¬ (p dvd coeff g 0 ∧ p dvd coeff h 0)
    by (metis const mult-dvd-mono power2-eq-square)
  then show False
    by (metis dg dh eisenstein-factor-helper fgh lg lh mid mult.commute prime)
qed
```

6.2 The polynomial and its real roots

The quintic, as an integer polynomial; it is irreducible by Eisenstein at $2::'a$.

```
definition PhiZ :: int poly
  where PhiZ ≡ [:2, −4, 0, 0, 0, 1:]

lemma degree-PhiZ: degree PhiZ = 5
  by (simp add: PhiZ-def)

lemma content-PhiZ: Polynomial.content PhiZ = 1
  by (simp add: PhiZ-def content-def)

lemma PhiZ-coeffs:
  coeff PhiZ 0 = 2 coeff PhiZ 1 = −4 coeff PhiZ 2 = 0
  coeff PhiZ 3 = 0 coeff PhiZ 4 = 0 coeff PhiZ 5 = 1
  by (simp-all add: PhiZ-def coeff-pCons numeral-eq-Suc)
```

A degree-0 factor of *PhiZ* is a unit.

```
lemma PhiZ-deg0-factor-unit:
  assumes eq: PhiZ = smult c q
  shows [:c:] dvd 1
proof −
  have normalize c dvd 1 using content-PhiZ
    by (metis eq content-smult dvd-def)
  then show [:c:] dvd 1 by (simp add: is-unit-const-poly-iff)
qed

theorem irreducible-PhiZ: irreducible PhiZ
proof (rule irreducibleI)
  show PhiZ ≠ 0 by (simp add: PhiZ-def)
  show ¬ PhiZ dvd 1 using degree-PhiZ by (auto dest!: dvd-imp-degree-le)
next
  fix a b
  assume eq: PhiZ = a * b
  show a dvd 1 ∨ b dvd 1
  proof (rule ccontr)
```

assume ¬ (*a dvd 1* ∨ *b dvd 1*)
then have §: *False* **if** *degree a = 0* ∨ *degree b = 0*
by (*metis degree-eq-zeroE that PhiZ-deg0-factor-unit eq mult-smult-right smult-one mult.commute*)
show *False*
proof (*rule eisenstein-no-factor*[*OF - eq - - -*])
show ⋀*i*. *i* < *degree PhiZ* ⟹ (*2*::*int*) *dvd coeff PhiZ i*
using *degree-PhiZ PhiZ-coeffs* **by** (*auto simp*: *less-Suc-eq numeral-eq-Suc*)
qed (*use PhiZ-coeffs degree-PhiZ* § **in** *force*)+
qed
qed

6.3 The real polynomial and its real roots

definition *Phi* :: *real poly*
where *Phi* ≡ [:*2*, −*4*, *0*, *0*, *0*, *1*:]

lemma *Phi-eval*: *poly Phi x* = *x*^*5* − *4*∗*x* + *2*
by (*simp add*: *Phi-def eval-nat-numeral algebra-simps*)

lemma *degree-Phi*: *degree Phi* = *5*
by (*simp add*: *Phi-def*)

At least three real roots, in disjoint intervals.

lemma *Phi-root-neg*: ∃ *x*. −*2* < *x* ∧ *x* < *0* ∧ *poly Phi x* = *0*
using *poly-IVT-pos*[*of* −*2 0 Phi*] **by** (*simp add*: *Phi-eval*)

lemma *Phi-root-01*: ∃ *x*. *0* < *x* ∧ *x* < *1* ∧ *poly Phi x* = *0*
using *poly-IVT-neg*[*of 0 1 Phi*] **by** (*simp add*: *Phi-eval*)

lemma *Phi-root-12*: ∃ *x*. *1* < *x* ∧ *x* < *2* ∧ *poly Phi x* = *0*
using *poly-IVT-pos*[*of 1 2 Phi*] **by** (*simp add*: *Phi-eval*)

The Sturm method counts the real roots: there are exactly three.

theorem *Phi-three-real-roots*: *card* {*x*::*real*. *poly Phi x* = *0*} = *3*
unfolding *Phi-def* **by** *sturm*

6.4 The complex roots: three real and one conjugate pair

The same polynomial over the complex numbers.

definition *PhiC* :: *complex poly* **where** *PhiC* ≡ [:*2*, −*4*, *0*, *0*, *0*, *1*:]

lemma *degree-PhiC*: *degree PhiC* = *5*
by (*simp add*: *PhiC-def*)

lemma *PhiC-nonzero*: *PhiC* ≠ *0*
by (*simp add*: *PhiC-def*)

lemma *PhiC-eval*: *poly PhiC x = x^5 − 4∗x + 2*
by (*simp add*: *PhiC-def eval-nat-numeral algebra-simps*)

lemma *pderiv-PhiC*: *pderiv PhiC = [:−4, 0, 0, 0, 5:]*
by (*simp add*: *PhiC-def pderiv-pCons*)

lemma *pderiv-PhiC-eval*: *poly* (*pderiv PhiC*) *x = 5∗x^4 − 4*
by (*simp add*: *pderiv-PhiC eval-nat-numeral algebra-simps*)

The quintic is squarefree: it shares no root with its derivative. (A common root would force $x = 5/8$, which does not satisfy $5x^4 = 4$.)

lemma *PhiC-rsquarefree*: *rsquarefree PhiC*
unfolding *rsquarefree-roots PhiC-eval pderiv-PhiC-eval*
by *algebra*

lemma *PhiC-roots-finite*: *finite* {*z*. *poly PhiC z = 0*}
using *PhiC-nonzero poly-roots-finite* **by** *blast*

Being squarefree of degree 5, it has exactly five distinct complex roots.

theorem *card-PhiC-roots*: *card* {*z*. *poly PhiC z = 0*} *= 5*
proof −
have *decomp*: *smult* (*lead-coeff PhiC*) (∏ *z* | *poly PhiC z = 0*. [:− *z*, *1*:]) *= PhiC*
using *PhiC-rsquarefree* **by** (*rule complex-poly-decompose-rsquarefree*)
have *lead*: *lead-coeff PhiC = 1* **by** (*simp add*: *PhiC-def*)
have *degree PhiC = degree* (∏ *z* | *poly PhiC z = 0*. [:− *z*, *1*:])
by (*metis decomp lead smult-1-left*)
also have ... = (∑ *z* | *poly PhiC z = 0*. *degree* [:− *z*, *1*:])
by (*simp add*: *degree-prod-eq-sum-degree*)
also have ... = *card* {*z*. *poly PhiC z = 0*} **by** *simp*
finally show *?thesis* **using** *degree-PhiC* **by** *simp*
qed

The real complex-roots are the images of the real roots, hence number three.

lemma *poly-PhiC-of-real*: *poly PhiC* (*complex-of-real x*) = *complex-of-real* (*poly Phi x*)
by (*simp add*: *PhiC-eval Phi-eval*)

lemma *real-roots-PhiC*: {*z*. *poly PhiC z = 0* ∧ *z* ∈ ℝ} = *complex-of-real* ‘ {*x*. *poly Phi x = 0*}
by (*auto simp*: *poly-PhiC-of-real elim*!: *Reals-cases*)

theorem *card-real-roots-PhiC*: *card* {*z*. *poly PhiC z = 0* ∧ *z* ∈ ℝ} *= 3*
by (*simp add*: *Phi-three-real-roots card-image inj-on-def real-roots-PhiC*)

Conjugation maps roots to roots (the coefficients are real).

lemma *cnj-poly-PhiC*: *poly PhiC* (*cnj z*) = *cnj* (*poly PhiC z*)
by (*simp add*: *PhiC-eval*)

lemma *cnj-root-PhiC*: *poly PhiC z = 0* ⟹ *poly PhiC* (*cnj z*) *= 0*

by (*simp add*: *cnj-poly-PhiC*)

Hence exactly two non-real roots.

theorem *card-nonreal-roots-PhiC*: *card* {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} = *2*
proof −
have *part*: {*z*. *poly PhiC z* = *0*} = {*z*. *poly PhiC z* = *0* ∧ *z* ∈ ℝ} ∪ {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ}
by *auto*
have {*z*. *poly PhiC z* = *0* ∧ *z* ∈ ℝ} ∩ {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} = {} **by** *auto*
then have *card* {*z*. *poly PhiC z* = *0*} = *card* {*z*. *poly PhiC z* = *0* ∧ *z* ∈ ℝ} + *card* {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ}
using *part PhiC-roots-finite* **by** (*metis* (*no-types*, *lifting*) *card-Un-disjoint finite-Un*)
then show *?thesis* **using** *card-PhiC-roots card-real-roots-PhiC* **by** *simp*
qed

lemma *cnj-notreal*: *z* ∉ ℝ ⟹ *cnj z* ∉ ℝ
by (*simp add*: *Reals-cnj-iff*)

lemma *cnj-ne-self-of-notreal*: *z* ∉ ℝ ⟹ *z* ≠ *cnj z*
by (*metis Reals-cnj-iff*)

The two non-real roots form a single conjugate pair, so complex conjugation transposes them while fixing the three real roots.

lemma *nonreal-roots-conj-pair*:
obtains *a* **where** {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} = {*a*, *cnj a*} **and** *a* ≠ *cnj a*
and *poly PhiC a* = *0* **and** *a* ∉ ℝ
proof −
obtain *a* **where** *a*: *poly PhiC a* = *0* **and** *anr*: *a* ∉ ℝ
using *card-nonreal-roots-PhiC that* **unfolding** *card-2-iff′* **by** *blast*
then have *sub*: {*a*, *cnj a*} ⊆ {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ}
using *cnj-notreal cnj-root-PhiC* **by** *blast*
have *ne*: *a* ≠ *cnj a*
by (*simp add*: *anr cnj-ne-self-of-notreal*)
have *card2*: *card* {*a*, *cnj a*} = *2* **using** *ne* **by** *simp*
have *fin*: *finite* {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} **using** *PhiC-roots-finite* **by** *simp*
have {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} = {*a*, *cnj a*}
using *sub card2 card-nonreal-roots-PhiC fin* **by** (*metis card-subset-eq*)
with *ne a anr* **show** *?thesis* **using** *that* **by** *blast*
qed

Summary of the root structure feeding the Galois-group argument.

theorem *PhiC-root-structure*:
card {*z*. *poly PhiC z* = *0*} = *5* ∧
card {*z*. *poly PhiC z* = *0* ∧ *z* ∈ ℝ} = *3* ∧
card {*z*. *poly PhiC z* = *0* ∧ *z* ∉ ℝ} = *2* ∧
(∀ *z*. *poly PhiC z* = *0* ⟶ *poly PhiC* (*cnj z*) = *0*)
using *card-PhiC-roots card-real-roots-PhiC card-nonreal-roots-PhiC cnj-root-PhiC*
by *blast*

6.5 Irreducibility of the quintic over the rationals

The complex quintic is the *of-int*-image of the integer quintic, equivalently the *of-rat*-image of its rational version.

lemma *PhiC-eq-map-of-int*: *PhiC* = *map-poly of-int PhiZ*
by (*simp add*: *PhiC-def PhiZ-def map-poly-pCons*)

lemma *map-of-rat-of-int-PhiZ*:
map-poly of-rat (*map-poly of-int PhiZ* :: *rat poly*) = (*PhiC* :: *complex poly*)
by (*simp add*: *PhiC-eq-map-of-int PhiZ-def map-poly-pCons*)

theorem *irreducible-over-Rats-PhiC*: *irreducible-over* (ℚ :: *complex set*) *PhiC*
proof −
have *degree-PhiZ-ne0*: *degree PhiZ* ≠ *0*
by (*simp add*: *degree-PhiZ*)
have *irreducible* (*map-poly of-int PhiZ* :: *rat poly*)
using *irreducible-PhiZ degree-PhiZ-ne0 content-PhiZ*
by (*metis Polynomial-Factorial.irreducible-int-imp-rat*)
moreover have *degree* (*map-poly of-int PhiZ* :: *rat poly*) > *0*
by (*simp add*: *degree-PhiZ degree-map-poly*)
ultimately show *?thesis*
using *irreducible-imp-irreducible-over-Rats map-of-rat-of-int-PhiZ* **by** *fastforce*
qed

6.6 The Galois group of the quintic is not solvable

The quintic has rational (indeed integer) coefficients.

lemma *PhiC-poly-over-Rats*: *PhiC* ∈ *poly-over* (ℚ :: *complex set*)
by (*simp add*: *PhiC-eq-map-of-int coeff-map-poly poly-over-iff-aux*)

Relabelling an involution that swaps exactly two points yields a transposition. Here τ is an involution of the finite set X enumerated by e, fixing every point except the pair $\{a, \tau\ a\}$.

lemma *pull-involution-two-points-transpose*:
fixes *e* :: *nat* ⇒ *complex* **and** *X* :: *complex set* **and** τ :: *complex* ⇒ *complex* **and** *n* :: *nat*
assumes *ebij*: *bij-betw e* {*0*..<*n*} *X*
and *tauX*: τ ‘ *X* = *X*
and *invol*: ⋀*x*. *x* ∈ *X* ⟹ τ (τ *x*) = *x*
and *a*: *a* ∈ *X* **and** *ta*: τ *a* ∈ *X* **and** *ane*: *a* ≠ τ *a*
and *fixed*: ⋀*x*. *x* ∈ *X* ⟹ *x* ≠ *a* ⟹ *x* ≠ τ *a* ⟹ τ *x* = *x*
defines *i* ≡ *inv-into* {*0*..<*n*} *e a* **and** *j* ≡ *inv-into* {*0*..<*n*} *e* (τ *a*)
shows *pull n e* (*restrict* τ *X*) = *Transposition.transpose i j*
proof
fix *k*
obtain *e*: *e i* = *a* *e j* = τ *a* **and** *ln*: *i* < *n j* < *n*
unfolding *i-def j-def* **using** *ebij a ta* **by** (*meson e-inv-e inv-e-lt*)
show *pull n e* (*restrict* τ *X*) *k* = *Transposition.transpose i j k*

```
  proof (cases k < n)
    case False
    then show ?thesis using ln by (simp add: pull-def transpose-def)
  next
    case True
    have ekX: e k ∈ X using ebij True by (auto simp: bij-betw-def)
    have pull n e (restrict τ X) k = inv-into {0..<n} e (τ (e k))
      using True ekX by (simp add: pull-def)
    also have ... = Transposition.transpose i j k
      using True a ebij e ekX fixed invol
      unfolding i-def j-def by (metis inv-e-e transpose-eq-iff)
    finally show ?thesis .
  qed
qed
```

6.7 The main theorem

The Galois group of the quintic $x^5 - 4x + 2$ — the group of automorphisms of its splitting subfield of *complex* fixing $\mathbb{Q}$ — is not solvable. This will imply that the quintic is not solvable by radicals.

```
theorem quintic-galois-group-not-solvable:
  defines X ≡ {z. poly PhiC z = 0}
  defines K ≡ generate-field (ℚ ∪ X)
  shows ¬ Group.solvable (field-auto K ℚ) (compose K) (identity K)
proof −
  — The base field and ambient setup.
  have sfQ: Subfield (ℚ :: complex set)
    using complex-subfield-Rats by (simp add: complex-subfield-iff-subfield)
  obtain finX: finite X and cardX: card X = 5
    using card-PhiC-roots by (fastforce simp: X-def)
  have FK: (ℚ :: complex set) ⊆ K unfolding K-def using subset-generate-field by
blast
  have XK: X ⊆ K unfolding K-def using subset-generate-field by blast
  have sfK: Subfield K unfolding K-def by (rule subfield-generate-field)
  have closed: ∃ b. poly q b = 0 if 0 < degree q for q :: complex poly
    using that by (intro fundamental-theorem-of-algebra) (simp add: constant-degree)
  — The Galois group acts on the roots.
  interpret GA: Group-Action field-auto K ℚ compose K identity K λσ. restrict σ X
X
    using FK PhiC-poly-over-Rats XK X-def finX galois-group-action sfK sfQ by blast
  — An enumeration of the five roots.
  obtain e where ebij: bij-betw e {0..<(5::nat)} X
    using ex-bij-betw-nat-finite[OF finX] cardX by force
  — Faithfulness: an automorphism fixing every root (and ℚ) is the identity on K.
  have Kgen: K = gen-subfield (ℚ ∪ X)
    by (simp add: K-def gen-subfield-eq-generate-field)
  have faith: GA.faithful
    unfolding GA.faithful-def
  proof (intro ballI impI)
```

fix σ **assume** *s*: $\sigma \in$ *field-auto K* $\mathbb{Q}$ **and** *triv*: $\forall x{\in}X.$ *restrict* σ *X x* = *x*
have *sgen*: $\sigma \in$ *field-auto* (*gen-subfield* ($\mathbb{Q} \cup X$)) $\mathbb{Q}$ **using** *s Kgen* **by** *simp*
have *complex-subfield K* **using** *sfK* **by** (*simp add*: *complex-subfield-iff-subfield*)
then have *allK*: $\forall x{\in}K.$ σ *x* = *x*
by (*metis Kgen field-auto-faithful restrict-apply s triv*)
have *sext*: $\sigma \in K \to_E K$ **using** *s* **by** (*simp only*: *field-auto-mem-iff*)
with *allK* **show** σ = *identity K*
by (*metis PiE-restrict restrict-ext*)
qed
— Finiteness of the Galois group: it embeds into the finite symmetric group on *X*.
have *finGal*: *finite* (*field-auto K* $\mathbb{Q}$)
proof −
have *inj-on* ($\lambda\sigma.$ *restrict* σ *X*) (*field-auto K* $\mathbb{Q}$)
using *GA.faithful-inj* [*OF faith*] *inj-on-restrict-eq* **by** *blast*
moreover have ($\lambda\sigma.$ *restrict* σ *X*) ' *field-auto K* $\mathbb{Q} \subseteq$ *transformations.Sym X*
using *GA.hom-closed* **by** *auto*
moreover have *finite* (*transformations.Sym X*)
by (*metis GA.mem-UnitsD finX finite-PiE finite-subset subsetI*)
ultimately show *?thesis* **by** (*metis inj-on-finite finite-subset*)
qed
— Transitivity: the orbit of any root is all of X.
have *trans*: $\exists s{\in}X.$ *GA.orbit s* = *X*
proof −
obtain *a* **where** *aX*: *a* ∈ *X* **using** *cardX* **by** (*metis card.empty equals0I zero-neq-numeral*)
have *surj*: $\forall b{\in}X.$ $\exists \sigma{\in}$*field-auto K* $\mathbb{Q}$. *restrict* σ *X a* = *b*
using *galois-action-transitive*[*OF sfQ finX PhiC-poly-over-Rats irreducible-over-Rats-PhiC - closed aX*]
unfolding *K-def X-def* **by** *blast*
have *X* ⊆ *GA.orbit a*
using *GA.orbit-mem-iff aX surj* **by** *force*
then show *?thesis* **using** *aX GA.orbit-subset* **by** *blast*
qed
— Complex conjugation contributes a transposition to the image.
have *transp*: $\exists\, i\ j.$ *i* < *5* ∧ *j* < *5* ∧ *i* ≠ *j*
∧ *Transposition.transpose i j* ∈ ($\lambda\sigma.$ *pull 5 e* (*restrict* σ *X*)) ' *field-auto K* $\mathbb{Q}$
proof −
have *cnjXX*: *cnj* ' *X* = *X*
unfolding *X-def* **using** *cnj-image-eq cnj-poly-PhiC* **by** *force*
obtain *a* **where** *Xnr*: {*z*. *poly PhiC z* = *0* ∧ *z* ∉ $\mathbb{R}$} = {*a*, *cnj a*} **and** *ane*: *a* ≠ *cnj a*
and *aroot*: *poly PhiC a* = *0* **and** *anr*: *a* ∉ $\mathbb{R}$ **by** (*rule nonreal-roots-conj-pair*)
have *aX*: *a* ∈ *X* **using** *aroot* **unfolding** *X-def* **by** *simp*
have *caX*: *cnj a* ∈ *X* **using** *aroot cnj-root-PhiC* **unfolding** *X-def* **by** *simp*
— cnj is the field automorphism that fixes $\mathbb{Q}$, and on *X* moves exactly *a*, *cnj a*.
have *cnjK*: *cnj* ' *K* = *K*
using *Kgen cnjXX splitting-subfield-cnj-closed* **by** *presburger*
have *QR*: ($\mathbb{Q}$ :: *complex set*) ⊆ $\mathbb{R}$
using *Reals-cnj-iff cnj-fixes-Rats* **by** *blast*
have *cnjaut*: *restrict cnj K* ∈ *field-auto K* $\mathbb{Q}$

using *FK QR cnjK cnj-field-auto complex-subfield-iff-subfield sfK* **by** *presburger*
— On X, cnj fixes everything except the conjugate pair.
have *fixed*: *cnj x = x* **if** *x ∈ X x ≠ a x ≠ cnj a* **for** *x*
using *Reals-cnj-iff X-def Xnr that* **by** *blast*
have *invol*: *cnj (cnj x) = x* **for** *x* **by** *simp*
— So the relabelling of cnj is a transposition.
define *i* **where** *i ≡ inv-into {0..<5} e a*
define *j* **where** *j ≡ inv-into {0..<5} e (cnj a)*
have *pull-t*: *pull 5 e (restrict cnj X) = Transposition.transpose i j*
unfolding *i-def j-def*
by (*rule pull-involution-two-points-transpose*[*OF ebij cnjXX invol aX caX ane fixed*])
obtain *5*: *i < 5 j < 5* **unfolding** *i-def j-def*
by (*meson ebij aX caX inv-e-lt*)
have *inej*: *i ≠ j*
unfolding *i-def j-def* **by** (*metis aX ane caX e-inv-e ebij*)
— This transposition is the image of the conjugation automorphism.
have *Transposition.transpose i j = pull 5 e (restrict (restrict cnj K) X)*
using *pull-t* **by** (*simp add*: *Int-absorb1 XK*)
then have *Transposition.transpose i j ∈ (λσ. pull 5 e (restrict σ X)) ‘ field-auto K* **Q**
using *cnjaut* **by** *blast*
then show *?thesis* **using** *5 inej* **by** *blast*
qed
— Assemble via the abstract Abel–Ruffini theorem (*p = (5::′a)*).
show *?thesis*
by (*rule GA.not-solvable-of-transitive-with-transposition*
[*OF ebij - - finGal faith trans transp*]) *auto*
qed

end

7 The quintic $x^5 - 4x + 2$ is not solvable by radicals

theory *Abel-Ruffini-Quintic*
imports
Radical-Normal-Closure — the forward direction: the normal-closure construction
Quintic-Example — concrete non-solvability of the quintic's Galois group
begin

The two halves of the argument meet here.

7.1 The main theorem

theorem *quintic-not-solvable-by-radicals*:
¬ *solvable-by-radicals* (**Q** :: *complex set*) {*r*. *poly PhiC r = 0*}
proof

```
  assume solv: solvable-by-radicals (ℚ :: complex set) {r. poly PhiC r = 0}
  — Names and standing facts.
  define X where X ≡ {r. poly PhiC r = 0}
  define Kq where Kq ≡ generate-field (ℚ ∪ X)
  have sfQ: Subfield (ℚ :: complex set)
    using complex-subfield-Rats by (simp add: complex-subfield-iff-subfield)
  have finX: finite X unfolding X-def by (rule PhiC-roots-finite)
  — Unpack the radical tower witnessing solvability.
  obtain Fs where tower: radical-tower (ℚ :: complex set) Fs
    and Xtop: {r. poly PhiC r = 0} ⊆ tower-top (ℚ :: complex set) Fs
    using solv by (auto simp: solvable-by-radicals-def)
  have Xtop': X ⊆ tower-top (ℚ :: complex set) Fs using Xtop by (simp add: X-def)
  have X-eq: X = {r. poly PhiC r = 0} by (simp add: X-def)

  have solvKq: Group.solvable (field-auto Kq ℚ) (compose Kq) (identity Kq)
    unfolding Kq-def
    by (rule radical-tower-root-field-solvable
      [OF sfQ tower PhiC-poly-over-Rats irreducible-over-Rats-PhiC
        X-eq finX Xtop' alg-closed-imp-poly-has-root])
  — Contradiction with the concrete non-solvability of the quintic Galois group.
  have ¬ Group.solvable (field-auto Kq ℚ) (compose Kq) (identity Kq)
    using quintic-galois-group-not-solvable
    unfolding Kq-def X-def by simp
  with solvKq show False by contradiction
qed

end
```

References

[1] M. Eberl. Sturm's theorem. *Archive of Formal Proofs*, 2014. https://isa-afp.org/entries/Sturm_Sequences.html, Formal proof development.

[2] T. mathlib Community. mathlib: `Mathlib.FieldTheory.AbelRuffini`. https://leanprover-community.github.io/mathlib4_docs/Mathlib/FieldTheory/AbelRuffini.html. Lean 4 / mathlib formalisation of the Abel–Ruffini theorem.