γ_{11}: Computational Certificates

THE MACADAMIAS 0001, 0014, 0015, 0059 AND 0052

Rémi Géraud-Stewart[1] and David Naccache[1,2]

[1] ENS, CNRS, PSL Research University, Paris, France
remi.geraud@ens.fr, david.naccache@ens.fr
[2] Fakultet inženjerskih nauka, Univerziteta u Kragujevcu
6 Sestre Janjić, 34000 Kragujevac, Serbia
david@kg.ac.rs

Abstract. The macadamias are the ten perfect forms of rank 9 for which the proof of the Hermite constant γ_{11} cannot be completed at rank 10. Four of them, 0001, 0014, 0059 and 0015, were treated at rank 11 by contact geometry in our previous work, and a fifth, 0052, by a gluing argument with a computer proof. To reduce dependency on pen-and-paper mathematical proofs and increase confidence in the announced results, this paper gives a second, machine-checked confirmation of these five cases. The new proof relies on computed certificates. A certificate counts only in *general mode*: it must hold for every lattice of rank nine that is admissibly marked by the form, and not only for the form itself. The certificate of a form is a tree of boxes over the space of its first gluings, every leaf of which is checked exactly, together with a ball of gluings around the rank-ten lattice stored with the form, on which some leaves rely. The ball comes from an exact theorem in which the gluing and the rank-nine member both move; the bound $\det L \geq 972$ follows from the inequality $\det L \geq \det M\,(4 - \rho^2(M))$ between a lattice L of rank 11, its primitive sublattice M of rank 10 and the covering radius $\rho(M)$. For 0059 the ball uses a contact-cone certificate of degree three; for 0015, whose stored lattice has no margin, a sharper localisation and an off-mean covering bound, both proved here; 0052 needs no ball. We state what each program checks, give the checking algorithms, all in integer and rational arithmetic, including a parallel final verification of a tree, and describe the publicly available database that stores the certificates, with the commands that check each form again from the database, the build and the reproduction pack. Floating point is used to find certificates but *never to accept them*. The five certificates check again, by a second route, the proofs of these five cases in our previous papers; they do not replace them.

Keywords: Hermite constant · Lattice packings · Covering radius · Delaunay subdivision · Exact certificates · Certificate database · Verified computation

1 Introduction

The root paper of the series [5] reduces the Hermite constant γ_{11} to a statement about the $4\,487\,813$ perfect forms of rank 9 that survive its selection: a lattice of rank 11 with minimum 4 and determinant below 972 would contain a chain $L \supset M \supset N$ of sections

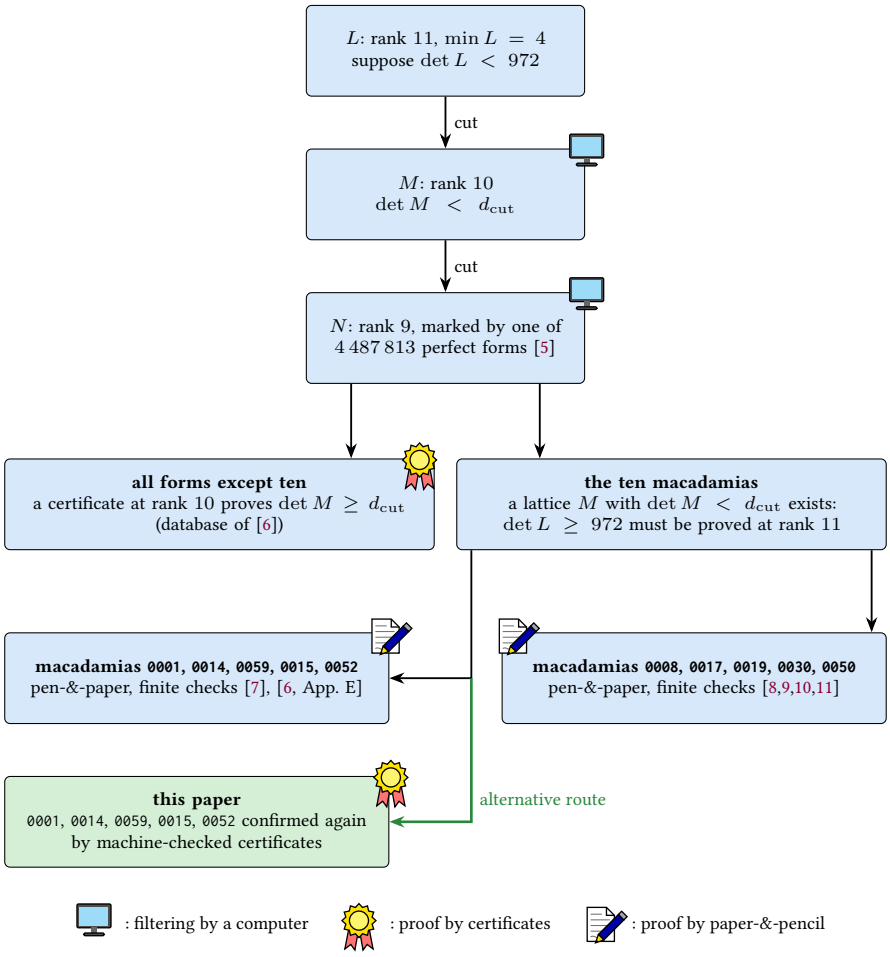


Fig. 1. Where this paper stands in the series. A lattice of rank 11 with minimum 4 and determinant below 972 would contain a chain $L \supset M \supset N$ [5]. For all forms except ten a certificate at rank 10 excludes the chain [6]. For the ten macadamias the bound must be proved at rank 11: for five of them in [7] and [6, Appendix E], which this paper confirms a second time by certificates, and for the other five in [8,9,10,11]. Blue: the earlier papers of the series; green: this paper.

of ranks 11, 10 and 9, and N would lie over one of these forms. For all but ten forms the database of [6] excludes the chain at rank 10, by an exact certificate that proves $\det M \geq d_{\text{cut}}$. The ten remaining forms, the *macadamias*, lie under an actual lattice of rank 10 with minimum 4 and determinant below d_{cut}, so no certificate at rank 10 can exist for them, and the bound $\det L \geq 972$ must be proved at rank 11. For four of them, `0001`, `0014`, `0059` and `0015`, this was done with pen and paper in the contact geometry paper [7]; for a fifth, `0052`, by the gluing argument and the computer proof of [6, Appendix E]. Figure 1 shows where this paper stands in the series.

Table 1. The five certificates. *Ball*: the ball at the stored gluing used by the tree, with its radius and the theorem that proves it.

form	ball	theorem	leaves
`0001`	$64497/200000$	Theorem 19	13696
`0014`	$249731/10^6$	Theorem 19	9670
`0059`	$343459/10^6$	Theorem 19, K of degree three (Lemma 10)	145815
`0015`	$251891/10^6$	Corollary 25 (Theorem 23, Lemma 21)	1463716
`0052`	none	leaves of the pack only	87470

What this paper does. It gives a *second, machine-checked confirmation* of these five cases. The confirmation uses the same toolbox as [7] (contact cones, the Delaunay cells of the stored lattice of rank 10 and its covering radius, and four lemmata of [6]), but it localises the gluing in another way, by a tree of boxes and a ball instead of the case-by-case geometry of [7], and it is checked in another way: every step by a program in exact arithmetic, instead of a human reader. The certificate of a form is a tree of boxes over the space of its first gluings, every leaf of which is checked exactly, together with the ball of first gluings on which some of its leaves rely. This paper describes *how these certificates are checked*, proves the theorems behind the checks, and describes how the certificates are stored and how a reader checks them again. For concision, we do not describe how certificates were found.

General mode. The rank-9 member N of a chain is not the form P itself: it is any lattice whose Gram matrix is admissibly marked by P (Definition 1). A certificate counts in this paper only if it holds for every such N. We call this *general mode*. A certificate that holds only for $N = P$ is a *fixed-N* certificate. It is a weaker and separate statement: it does not exclude the chains over P, and no result of this paper uses one.

Three things to keep in mind.

1. This is a *second, machine-checked* confirmation. The proofs of [7] and [6, Appendix E] remain the proofs of record, and no result of theirs is used here: the inequalities that we need are quoted from the common toolbox of [6], where they are proved, or proved here (Sections 2 and 3).
2. The five certificates check again, by a second route, the proofs of [7] for `0001`, `0014`, `0059` and `0015`, and the computer proof of [6, Appendix E] for `0052` (Table 8 gives their lengths). They *double* these proofs; they do not replace them.
3. Every decision of every check is made in exact integer or rational arithmetic. Floating point was used to *find* certificates, never to *accept* them; where a check uses a floating-point number at all (Section 4.1), it can only make the check fail.

The result. For each of the concerned forms, the exact verification of its tree in general mode is valid, with the ball of the form where the tree uses one (Computational result 30); by Proposition 27, every chain over the form has $\det M \geq d_{\mathrm{cut}}$ or $\det L \geq 972$ (Theorem 32). Table 1 gives the certificates.

Organisation. Section 2 recalls the chains, the gluing space and the inequalities that every check relies upon. Section 3 defines the certificates and proves the theorems behind the balls: the general simple ball (Section 3.3), with contact-cone certificates of degree two, three and four, and the ball at a stored lattice without margin (Section 3.4). Section 4 gives the checking algorithms, including the parallel final verification of a tree. Section 5 describes the database. Section 6 explains how a reader re-checks a form from the archive. Section 7 states what is proved and on what it rests. Appendix A reports the restriction of an integer test and its extension; no published result is affected. Appendix B gives the layout of the stored files, and Appendix C the algorithms of the checkers.

2 Chains over a form

2.1 The objects

We use the notation of [6, Section 2]. A form is named by its record `file:row` in the data set of [3]; its record gives an integral matrix G with $s = \min G$. We write $P_{\mathrm{rec}} = \frac{4}{s}G$ for the form in the basis of its record, in which the balls are checked, and P for the form in the basis U named by its certificate, in which the tree is written:

$$P = UP_{\mathrm{rec}}U^{\mathsf{T}} = \frac{4}{s}\,UGU^{\mathsf{T}}, \qquad \min P = 4.$$

The vectors $C = \operatorname{Min} P$ (one per pair $\pm c$) are recomputed exactly from UGU^{T} by every check. Let

$$d_{\mathrm{cut}} = 4\,\sqrt[11]{972^9} = 1113.048\ldots,$$

the threshold that the programs call `dmax`. A rational upper bound $\overline{d}_{\mathrm{cut}} \geq d_{\mathrm{cut}}$ with relative error below 10^{-15} is computed in integers, as the least multiple of 10^{-15} whose eleventh power is at least $4^{11}972^9$, and a test $d < d_{\mathrm{cut}}$ is decided exactly by comparing d^{11} with $4^{11}972^9$.

Definition 1 (admissible marking; chains over a form). *Let P be a form of rank 9 with $\min P = 4$, and let C be its set of minimal vectors, one per pair $\pm c$. A positive definite 9×9 matrix Q is* admissibly marked by P *[5, Section 6.2] if $\min Q \geq 4$ and*

$$X := Q^{-1} = \sum_{c \in C} \lambda_c\, cc^{\mathsf{T}}, \qquad 0 \leq \lambda_c \leq \tfrac{1}{4}, \qquad \sum_{c \in C} \lambda_c \leq \tfrac{9}{4}. \tag{1}$$

For such a Q, $\tau \in \mathbb{R}^9$ and $u > 0$, the chain matrix

$$G(Q, \tau, u) = \begin{pmatrix} Q & Q\tau \\ \tau^{\mathsf{T}}Q & Q[\tau] + \frac{1}{u} \end{pmatrix} \tag{2}$$

is the Gram matrix of the lattice $M = N \oplus \mathbb{Z}g$ of rank 10 in which N has Gram matrix Q and g has projection τ on $N \otimes \mathbb{R}$ (in the basis of N) and squared distance $1/u$ to it. A chain over P *is a chain $L \supset M \supset N$ of the kind that [5, Proposition 15] produces,*

without its inequalities: L of rank 11 with $\min L \geq 4$, $M = L \cap n_{11}^{\perp}$ and $N = M \cap n_{10}^{\perp}$ for shortest vectors n_{11} of L^* and n_{10} of M^*, in which N has a Gram matrix Q admissibly marked by P. In that basis of N the Gram matrix of M is $G(Q, \tau, u)$ with $u = |n_{10}|^2 = \min M^*$. We write $d = \det M$, $\Delta = \det L$, and $G(\tau, u) = G(P, \tau, u)$ for the chain matrices with $Q = P$ (the stored lattice and the centres of the balls). The class of τ modulo $\mathbb{Z}^9$, up to sign, is the first gluing *of the chain.*

Figure 2 shows a chain. These are exactly the chains of [5, Section 6.2] and of [7]: the rank-9 member of a chain of [5, Proposition 15] that lies over P has a Gram matrix admissibly marked by P, and $Q = P$ is the special case of the stored lattice. The leaves of the reproduction pack use the conditions of [6, Section 2.4], and so do the ball theorems below. A statement about all these chains is in general mode; a statement about the chains with $Q = P$ only is a fixed-N statement. With $X = Q^{-1}$ the conditions are:

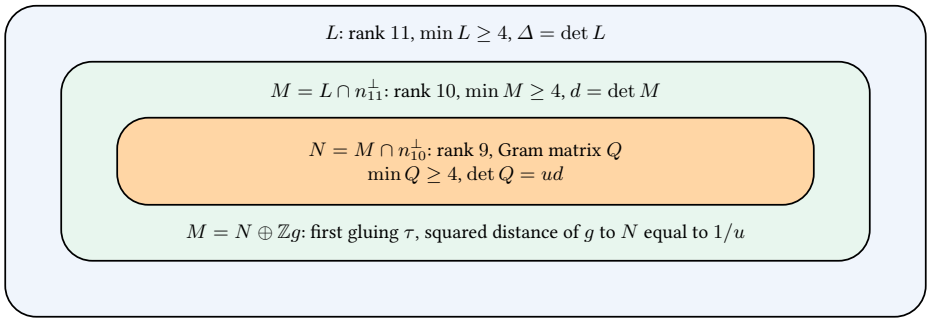
- **(M)** $\min M \geq 4$: $G(Q, \tau, u)[(x, k)] = Q[x + k\tau] + k^2/u \geq 4$ for $(x, k) \in \mathbb{Z}^{10} \setminus \{0\}$; and $\min Q \geq 4$, so $Q[c] \geq 4 = P[c]$ for every $c \in C$;
- **(L2)** $\min M^* = u$: $X[z] + u\,(k - z \cdot \tau)^2 \geq u$ for $(z, k) \in \mathbb{Z}^{10} \setminus \{0\}$; in particular $X[z] \geq u\,(1 - \|z \cdot \tau\|^2)$ for $z \neq 0$;
- **(Mk)** $\operatorname{tr}(XP) = \sum_c \lambda_c P[c] = 4 \sum_c \lambda_c \leq 9$, by (1);
- **(u)** $u \leq \sqrt[5]{4/3}$, tested exactly as $3u^5 \leq 4$;

The first three follow from Definition 1; **(u)** does not: it comes from the value of γ_{10} [5, Theorem 3], applied to the dual of M [6, Section 2.4]. In [6] the condition **(Mk)** is a consequence of its condition (Domain). There remain, for the extension, $\min L \geq 4$ and the primitivity of M in L. The Schur complement of Q in (2) is $1/u$, so

$$d = \det G(Q, \tau, u) = \frac{\det Q}{u}. \tag{3}$$

Lemma 2 (images of a chain). *Let $k \in \mathbb{Z}^9$ and let g be an integer matrix with $|\det g| = 1$ and $g^{\mathsf{T}} P g = P$. The chain matrices $G(Q, \tau, u)$, $G(Q, \tau + k, u)$, $G(Q, -\tau, u)$ and $G(g^{-\mathsf{T}} Q g^{-1}, g\tau, u)$ are Gram matrices of one and the same lattice in four bases, and $g^{-\mathsf{T}} Q g^{-1}$ is admissibly marked by P if and only if Q is. Hence the chains over P with first gluings τ, $\tau + k$, $-\tau$ and $g\tau$ correspond to one another, with the same d, the same extensions L and the same values of Δ. Moreover, for every integer matrix U with $|\det U| = 1$, $G(UQU^{\mathsf{T}}, U^{-\mathsf{T}}\tau, u)$ is a Gram matrix of the same lattice in a fifth basis, UQU^{T} is admissibly marked by UPU^{T} if and only if Q is admissibly marked by P, and $(UPU^{\mathsf{T}})[U^{-\mathsf{T}}\delta] = P[\delta]$ for every $\delta \in \mathbb{R}^9$. Hence the chains over P_{rec} with first gluing τ are the chains over $P = UP_{\mathrm{rec}}U^{\mathsf{T}}$ with first gluing $U^{-\mathsf{T}}\tau$, and a ball (τ_c, r) of P_{rec} (Section 3.2) is the ball $(U^{-\mathsf{T}}\tau_c, r)$ of P.*

Proof. With $T = \left(\begin{smallmatrix} I & k \\ 0 & 1 \end{smallmatrix}\right)$, $T' = \left(\begin{smallmatrix} I & 0 \\ 0 & -1 \end{smallmatrix}\right)$ and $T'' = \left(\begin{smallmatrix} g & 0 \\ 0 & 1 \end{smallmatrix}\right)$, all in $GL_{10}(\mathbb{Z})$, a direct computation gives $T^{\mathsf{T}} G(Q, \tau, u) T = G(Q, \tau + k, u)$, $T'^{\mathsf{T}} G(Q, \tau, u) T' = G(Q, -\tau, u)$ and $T''^{\mathsf{T}} G(Q', g\tau, u) T'' = G(g^{\mathsf{T}} Q' g, \tau, u)$ for every Q'; take $Q' = g^{-\mathsf{T}} Q g^{-1}$. Since $P[gc] = (g^{\mathsf{T}} P g)[c] = 4$, the map $c \mapsto gc$ permutes C up to signs, so $Q'^{-1} = gXg^{\mathsf{T}} = \sum_c \lambda_c\,(gc)(gc)^{\mathsf{T}}$ has the form (1) with the same weights; and $\min Q' = \min Q$. The other direction is the same with g^{-1}. For the change of basis, with $T_U = \left(\begin{smallmatrix} U & 0 \\ 0 & 1 \end{smallmatrix}\right)$,



n_{11} is a shortest vector of L^*; n_{10} is a shortest vector of M^* and $u = |n_{10}|^2 = \min M^*$. The Gram matrix Q of N is *admissibly marked* by the form P: $\min Q \geq 4$ and $Q^{-1} = \sum_c \lambda_c\, cc^{\mathsf{T}}$ with $0 \leq \lambda_c \leq \frac{1}{4}$ and $\sum_c \lambda_c \leq \frac{9}{4}$, where c runs over one vector of each opposite pair of $\mathrm{Min}\, P$. What must be shown: every such chain has $d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$.

Fig. 2. A chain $L \supset M \supset N$ over a form P (Definition 1; after [5, Figure 11]). N has a Gram matrix Q admissibly marked by P, $M = N \oplus \mathbb{Z}g$ is given by Q, its first gluing τ and u, and L is any lattice of rank 11 with minimum at least 4 in which M is primitive.

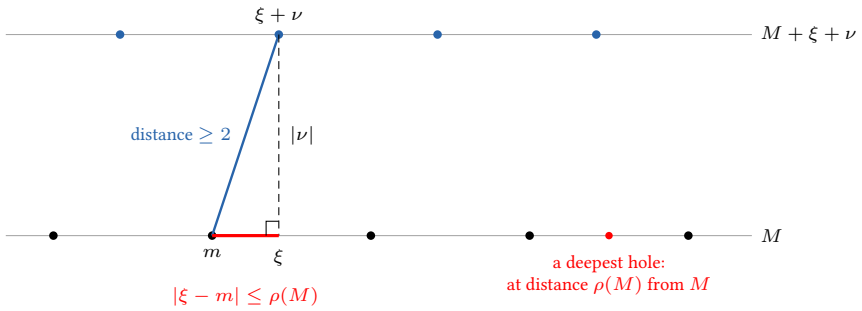
$T_U G(Q, \tau, u) T_U^{\mathsf{T}} = G(UQU^{\mathsf{T}}, U^{-\mathsf{T}}\tau, u)$; the minimal vectors of UPU^{T} are the vectors $U^{-\mathsf{T}}c$, $c \in C$, and $(UQU^{\mathsf{T}})^{-1} = \sum_c \lambda_c\, (U^{-\mathsf{T}}c)(U^{-\mathsf{T}}c)^{\mathsf{T}}$ with the same weights; the minimum and the determinant do not change. □

The integer matrices g with $g^{\mathsf{T}} P g = P$ form the automorphism group of P in the convention of the programs (columns); [6, Section 5.8] writes the same group with rows, $V = g^{\mathsf{T}}$.

The gluing space. By Lemma 2, everything depends on τ only through its class in $\mathbb{R}^9/\mathbb{Z}^9$ up to sign, and only up to the automorphisms of P. A tree of boxes [6, Section 5.7] parametrises this space as follows. The head of a tree carries a *pool*, a list of nonzero vectors z_i of $\mathbb{Z}^9$ that its leaves use, and its *triangles*, triples of pool vectors with $z_i \pm z_j \pm z_k = 0$ for some choice of the signs. Nine vectors of the pool form a basis of $\mathbb{Z}^9$ (`"box_basis"`); M_i is the integer inverse of their matrix, $L_{\mathrm{box}} = L_{\mathrm{int}}/D_L$ a lower triangular rational matrix with ones on the diagonal (`"box_L"`, `"box_DL"`), and

$$\tau = \frac{M_i L_{\mathrm{int}}\, t}{UU}, \qquad UU = 4096\, D_L, \tag{4}$$

where $t \in \mathbb{R}^9$ is the coordinate of a point of a box, in units of $1/4096$ (the corners of the boxes are integers). The root box $[0, 2048] \times [0, 4096]^8$ contains a representative of every first gluing [6, Section 5.7]; when the certificate carries symmetry data, the root box is the box `"sym.root"`, which contains a *fundamental domain*, that is a set that meets the orbit of every first gluing under the automorphisms of P and the translations by $\mathbb{Z}^9$, and the leaves of kind `outside` close the boxes that contain no point of that domain [6, Proposition 15]. The map $t \mapsto \tau$ is affine, so a function of τ that is convex reaches its maximum over a box at one of its 512 vertices.



$$4 \;\leq\; |\xi - m|^2 + |\nu|^2 \;\leq\; \rho^2(M) + |\nu|^2, \qquad \det L = \det M \cdot |\nu|^2 \;\geq\; \det M\,(4 - \rho^2(M))$$

Fig. 3. Lemma 3 in a picture. L is the union of the layers $M + k(\xi + \nu)$, $k \in \mathbb{Z}$, where ξ lies in the span of M and ν is orthogonal to it. The point $\xi + \nu$ of the next layer is at distance at least 2 from the point m of M nearest to ξ, and ξ is within the covering radius $\rho(M)$ of m; so $|\nu|^2 \geq 4 - \rho^2(M)$. The picture shows M as a line; in the paper it has rank 10.

2.2 The stored lattice of a macadamia

The row of a macadamia in the certificate database of the reproduction pack (the row "Mathematical Proof" of [6, Section 6]) records a chain matrix $G_0 = G(\tau_0, u_0)$ over the form: rational τ_0 and u_0 with $\min G_0 = 4$ and $d_0 = \det G_0 < d_{\text{cut}}$. We call M_0, the lattice of Gram matrix G_0, the *stored lattice* and τ_0 the *stored gluing*. The stored lattice is the reason why no certificate at rank 10 exists for the form, and it is the centre of the simple ball (Section 3.3). Every check reads G, s, τ_0 and u_0 from that row. The files of a ball repeat P, τ_0 and u_0, and the check of the ball tests that they equal those of the row (Algorithms 6 and 7). The registry of balls (Section 4.5) computes P from the row; it takes the centre and the radius from the verified ball file, after testing again that the P of that file is the P of the row.

2.3 The inequalities behind every ball

Lemma 3 (a primitive extension; [6, Lemma 41]). *Let L have rank 11 and minimum at least 4, and let $M \subset L$ be primitive of rank 10. Then*

$$\Delta = \det L \;\geq\; \det M\,\big(4 - \rho^2(M)\big),$$

where $\rho(M)$ is the covering radius of M in its span.

The names of the programs (`112*.py`) and the label "Lemma 12" in their outputs refer to Lemma 3. Its proof is in [6, Lemma 41]. Figure 3 shows the argument.

Lemma 4 (averaging in a cell; [6, Lemma 42]). *Let a_i be finitely many points, $\pi_i \geq 0$, $\sum_i \pi_i = 1$ and $\sum_i \pi_i a_i = x$. For every positive definite G, $\min_i G[a_i - x] \leq \sum_i \pi_i G[a_i] - G[x]$.*

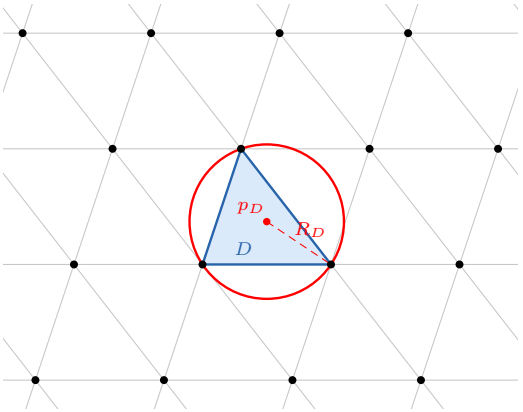


Fig. 4. A Delaunay cell of a lattice, in the plane. The cell is the convex hull of the lattice points on an empty sphere; the cells tile the space face to face, and the squared covering radius is the largest R_D^2 (Lemma 5). The balls of the paper use the Delaunay cells of a lattice of rank 10: 4356 to 7302 cells up to translations and sign.

A *Delaunay cell* of a lattice is the convex hull of the lattice points on a sphere whose interior contains no lattice point; the full-dimensional cells tile space, face to face [1,2,13]. A cell is determined by the centre of its sphere, and its vertices are *all* the lattice points on the sphere. Figure 4 shows a cell in the plane.

Lemma 5 (covering radius from the Delaunay cells). *The squared covering radius of a lattice is the largest squared radius of its Delaunay cells. More generally, let a positive definite metric G be given in coordinates, and let the Delaunay cells of a reference lattice Λ (for another metric G_0) have centres p_D, vertex sets V_D and radii R_D. If, for every cell D, $G[v - p_D] \le h_D$ for every $v \in V_D$, then $\rho_G^2(\Lambda) \le \max_D h_D$.*

Proof. A point y lies in a cell D; write it as a convex combination $y = \sum_v \lambda_v v$ of the vertices of D. By Lemma 4 with $a_v = v - p_D$, $\min_v G[v - y] \le \sum_v \lambda_v G[v - p_D] - G[y - p_D] \le h_D$. For $G = G_0$ and $h_D = R_D^2$ this gives $\rho^2 \le \max_D R_D^2$, and the centre of a cell of largest radius is at distance at least R_D from every lattice point. □

2.4 A chain near a centre

A *centre* is a pair (τ_c, u_c), $\tau_c \in \mathbb{Q}^9$, $u_c \in \mathbb{Q}_{>0}$, with the chain matrix $G_c = G(\tau_c, u_c)$ over the form itself and $d_c = \det G_c = \det P/u_c$. For a symmetric 10×10 matrix E we write

$$t(E) = \operatorname{tr}(G_c^{-1}E), \qquad \widehat{E} = G_c^{-1/2} E G_c^{-1/2}.$$

Lemma 6 (the identities of a chain near a centre). *For a chain over P with chain matrix $G = G(Q, \tau, u)$ let $\delta = \tau - \tau_c$, $v = u_c/u$, $E = G - G_c$, $X = Q^{-1}$, $S = P^{-1/2} Q P^{-1/2}$ and $s_N = \operatorname{tr} S - 9 = \operatorname{tr}(P^{-1}Q) - 9$. Then*

a $d = d_c \det S\, v$, *and* $\det(I + \widehat{E}) = \det(G_c^{-1}G) = d/d_c$;

b $t(E) = s_N + u_c\,Q[\delta] + v - 1$;
c $\operatorname{tr}(G_c G^{-1}) - 10 = \big(\operatorname{tr}(XP) - 9\big) + u\,P[\delta] + \frac{1}{v} - 1$.

Proof. a is (3) for G and for G_c. With

$$G_c^{-1} = \begin{pmatrix} P^{-1} + u_c\tau_c\tau_c^{\mathsf T} & -u_c\tau_c \\ -u_c\tau_c^{\mathsf T} & u_c \end{pmatrix} \qquad \text{and} \qquad G^{-1} = \begin{pmatrix} X + u\tau\tau^{\mathsf T} & -u\tau \\ -u\tau^{\mathsf T} & u \end{pmatrix},$$

block multiplication gives

$$\begin{aligned} \operatorname{tr}(G_c^{-1}G) &= \operatorname{tr}(P^{-1}Q) + u_c\big(Q[\tau] - 2\tau_c^{\mathsf T}Q\tau + Q[\tau_c]\big) + u_c/u, \\ \operatorname{tr}(G_c G^{-1}) &= \operatorname{tr}(PX) + u\big(P[\tau] - 2\tau_c^{\mathsf T}P\tau + P[\tau_c]\big) + u/u_c, \end{aligned}$$

which are b and c. □

2.5 What must be shown

For a form P we must show:

$$\text{every chain over } P \text{ has } d \ge d_{\mathrm{cut}} \text{ or } \Delta \ge 972. \tag{E$_P$}$$

A chain with $d \ge d_{\mathrm{cut}}$ contradicts the chain proposition of the root paper [5, Proposition 15], so (E$_P$) excludes the chains with $\Delta < 972$ over P, which is the aim.

3 Certificates

3.1 Trees of boxes and their leaves

Definition 7 (certificate of a form). *The* certificate of a form *consists of*
a a tree of boxes *in the format of the reproduction pack [6, Section 5.7]: a head (the basis U, the pool, `"box_basis"`, `"box_L"`, `"box_DL"`, `"unit"`, the number of leaves, and possibly symmetry data) and a binary tree whose inner nodes $\{$`"v"`$: j,$ `"at"`$: x,$ `"le"`, `"ge"`$\}$ cut a box by $t_j \le x$ and $t_j \ge x$, and whose leaves carry a certificate valid on their box;*
b *the* balls *on which its basin leaves rely (Sections 3.2–3.4).*

Figure 5 shows the parts of a certificate and the algorithm that checks each.
A *ball* of a form is a pair (τ_c, r), $\tau_c \in \mathbb{Q}^9$, $r \in \mathbb{Q}_{>0}$, with the property

$$\text{every chain over } P \text{ with } P[\tau - \tau_c] \le r^2 \text{ has } d \ge d_{\mathrm{cut}} \text{ or } \Delta \ge 972. \tag{B}$$

Two families of leaves occur. The *leaves of the reproduction pack* are those of [6, Table 1]: the kinds `box`, `pattern`, `pspec` and `uspec` prove $d \ge d_{\mathrm{cut}}$ on their box, the kind `three` proves $\Delta \ge 972$ on its box whatever the *second gluing* (the class modulo $\mathbb{Z}^{10}$ that places L over M, as the first gluing places M over N [6, Section 2.4]), and the

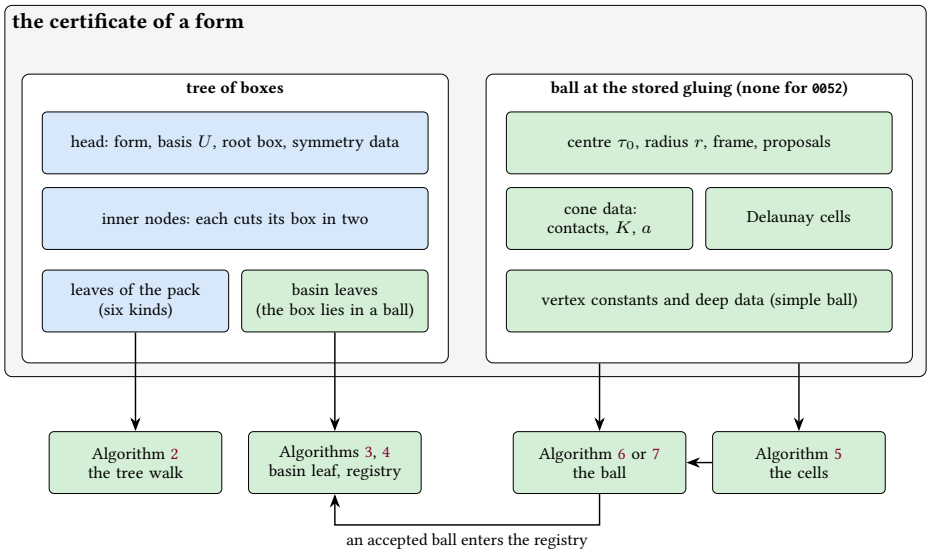


Fig. 5. The parts of a certificate and the algorithm that checks each. The tree is walked by the verifier of the reproduction pack (Algorithm 2), with one more branch for the basin leaves (Algorithm 3), which rely on the ball; the ball is checked from its own data (Algorithm 6 or 7), its Delaunay cells by Algorithm 5, and it enters the registry (Algorithm 4) only when accepted.

kind `outside` proves that its box contains no point of the fundamental domain. A *basin leaf* is one more kind, written in one of two ways:

$$\{\texttt{"H"}\colon 0,\ \texttt{"kind"}\colon\texttt{"gball"},\ \texttt{"gc"}\colon c,\ \texttt{"gk"}\colon [k_1,\dots,k_9],\ \texttt{"gr2"}\colon [p,q]\},$$

or, equivalently, with the keys `"basin"`, `"bc"`, `"bk"`, `"br2"`; it claims that its box lies in the ball number c of the form, translated by the integer vector k, with squared radius p/q at most the certified one (Section 4.4). The balls are those of Theorem 19 and Corollary 25, so a basin leaf holds for every chain over P, whatever the admissibly marked rank-9 member. Table 2 lists the kinds and their checkers.

Lemma 8 (tree argument; [6, Lemma 11 and Proposition 18]). *Suppose the leaves of a tree of boxes cover its root box, every leaf of the reproduction pack verifies on its box, and every basin leaf lies in a ball, that is a pair (τ_c, r) with the property (B). Then (E_P) holds.*

Proof. The argument of [6, Lemma 11 and Proposition 18], extended to the basin leaves. A chain over P has a first gluing with a representative in the root box (Lemma 2 and [6, Section 5.7 and Proposition 15]), hence in a leaf. An `outside` leaf contains no such representative. A leaf of the reproduction pack gives $d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$ for every chain whose first gluing lies in its box; so does a basin leaf, by (B). □

Table 2. The leaves of a certificate, what each proves on its box, and the function of `k11lib.py` or of this paper that checks it. All but the last row are leaves of the reproduction pack [6, Table 1], those that occur in the five trees; the last row is the leaf added by this paper.

kind	proves on its box	checked by
`box`	$d \geq d_{\mathrm{cut}}$	`box_leaf_value`
`pattern`	$d \geq d_{\mathrm{cut}}$, on the ranges of the box	`bb_leaf_value`
`pspec`	$d \geq d_{\mathrm{cut}}$	`pspec_leaf_value`
`uspec`	$d \geq d_{\mathrm{cut}}$, piecewise in u	`uspec_leaf_pair`
`three`	$\Delta \geq 972$, for every second gluing	`three_leaf_value`
`outside`	no point of the fundamental domain	`sym_outside_value`
`gball`, `basin`	$d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$ (the box lies in a general ball)	`gball_leaf_exact`, Algorithm 3

3.2 Balls

A ball of a form was defined in Section 3.1 by the property (B). The chains of (B) are all the chains of Definition 1: their rank-9 member is any lattice with a Gram matrix admissibly marked by P, not only P itself. By Lemma 2, if (τ_c, r) satisfies (B), so do $(\tau_c + k, r)$, $(-\tau_c, r)$ and $(g\tau_c, r)$ for $k \in \mathbb{Z}^9$ and every automorphism g of P: these maps are isometries for the metric P and send chains over P to chains over P with the same d and Δ. Figure 6 shows a tree of boxes with balls.

3.3 Simple balls: the general theorem

A simple ball is centred at a *cone centre*: a centre (τ_c, u_c) with $\min G_c = 4$ whose minimal vectors are eutactic; the stored gluing, with $G_0 = G(\tau_0, u_0)$ and $d_0 = \det P/u_0$, is the cone centre used in this paper. The theorem uses two contact cones: that of G_c, of rank 10, for the lattice M, and that of P, of rank 9, for its member N.

Contact cones. Let G be positive definite of size n and C_G a set of integer vectors with $G[c] = 4$ (the *contacts*). Its *contact cone* is $\mathcal{K}(G) = \{E \in \mathrm{Sym}_n : E[c] \geq 0 \text{ for every } c \in C_G\}$; for E in it write $t(E) = \mathrm{tr}(G^{-1}E)$ and $\widehat{E} = G^{-1/2}EG^{-1/2}$. By **(M)**, $G(Q, \tau, u) - G_c \in \mathcal{K}(G_c)$ for every chain over P and every cone centre (since $G(Q, \tau, u)[c] \geq 4 = G_c[c]$), and $Q - P \in \mathcal{K}(P)$ for every admissibly marked Q (with $C_P = C$).

Lemma 9 (constants of a contact cone). *Suppose that $G^{-1} = \sum_{c \in C_G} \varrho_c\, cc^{\mathsf{T}}$ with $\varrho_c \geq 0$ (eutaxy). Then for every $E \in \mathcal{K}(G)$:*

a *$t(E) = \sum_c \varrho_c E[c] \geq 0$;*

b *let $\tilde{K}$ and $\alpha_{cc'} \geq 0$ be rational, let $R(E) = \tilde{K}\, t(E)^2 - \|\widehat{E}\|_F^2 - \sum_{c \leq c'} \alpha_{cc'} E[c]E[c']$, a quadratic form with coefficients R_{pq} in the coordinates E_{ij} $(i \leq j)$, and let $\psi_R = \left(\sum_{p,q} |R_{pq}|\right) g(G)^2$, where $g(G)$ is the Gershgorin bound of the largest eigenvalue of G; if $\psi_R < 1$, then $\|\widehat{E}\|_F^2 \leq K\, t(E)^2$ for every $K \geq \tilde{K}/(1 - \psi_R)$;*

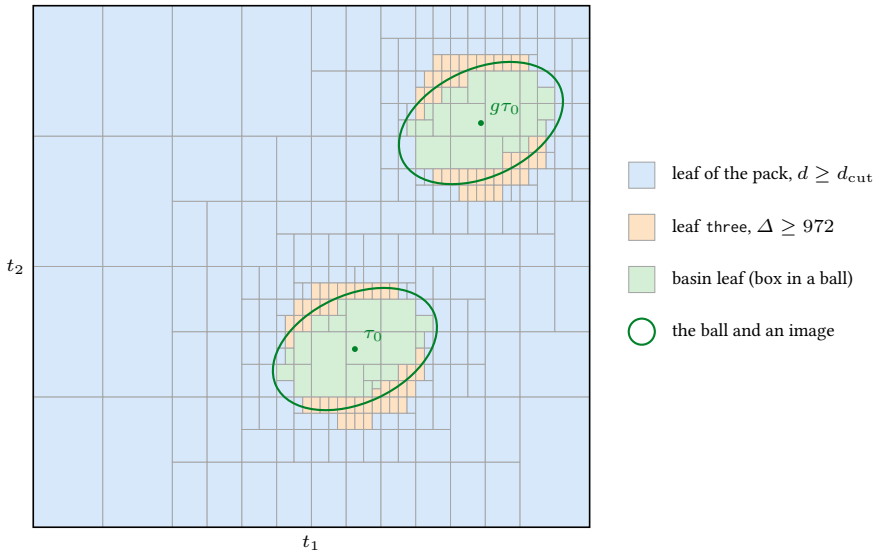


Fig. 6. A tree of boxes with basin leaves, in a schematic two-dimensional slice of the gluing space (the real space has dimension 9, and the trees of Table 7 have between 9670 and 1463716 leaves). Every ball is an ellipse of the same shape, because the metric P is the same everywhere. A box is a basin leaf when its vertices lie in one ball; the ball surrounds the stored gluing τ_0 and its images.

c *if* $\|\widehat{E}\|_F^2 \leq K\, t(E)^2$, *then* $\lambda_{\min}(\widehat{E}) \geq -a\, t(E)$ *and* $\lambda_{\max}(\widehat{E}) \leq b\, t(E)$, *where* $a = (s-1)/n$ *and* $b = (s+1)/n$ *for any* $s \geq \sqrt{1 + n((n-1)K - 1)}$.

Proof. a is the eutaxy identity applied to E. b: on $\mathcal{K}(G)$ the sum of the α terms is nonnegative, so $\tilde{K}t^2 - \|\widehat{E}\|_F^2 \geq R(E) \geq -\psi_R \|\widehat{E}\|_F^2$, because every entry of $E = G^{1/2}\widehat{E}G^{1/2}$ is at most $g(G)\|\widehat{E}\|_F$ in absolute value; hence $\|\widehat{E}\|_F^2 \leq \tilde{K}t^2/(1-\psi_R)$. c: the n eigenvalues of $\widehat{E}$ have sum t and sum of squares at most Kt^2; if λ is one of them, the $n-1$ others have sum $t-\lambda$, so $\lambda^2 + (t-\lambda)^2/(n-1) \leq Kt^2$, which gives $-at \leq \lambda \leq bt$. □

The identity behind b has degree two in E. The same constant K can come from an identity of degree three or four, which is often much better.

Lemma 10 (certificates of degree three and four). *Let G and C_G be as in Lemma 9 (eutaxy), and write $\ell_c(E) = E[c]$. Let K and the numbers $\beta \geq 0$ be rational, and suppose that one of the following two identities between polynomials in the entries of E holds:*

$$K\, t(E)^3 - \|\widehat{E}\|_F^2\, t(E) = \sum_{c_1 \leq c_2 \leq c_3} \beta_{c_1 c_2 c_3}\, \ell_{c_1}(E)\, \ell_{c_2}(E)\, \ell_{c_3}(E), \tag{5}$$

$$K\, t(E)^4 - \|\widehat{E}\|_F^2\, t(E)^2 = \sum_{c_1 \leq c_2 \leq c_3 \leq c_4} \beta_{c_1 c_2 c_3 c_4}\, \ell_{c_1}(E)\, \ell_{c_2}(E)\, \ell_{c_3}(E)\, \ell_{c_4}(E), \tag{6}$$

Table 3. Constants K of the contact cone of the stored lattice M_0 (rank 10), by the degree of the identity that gives them. The identities of degree three and four are verified exactly, by `l12k11.cubic_identity_ok` and `l12k11.quartic_identity_ok`. The constants of degree two are the certified ones of Lemma 9 (b), rounded (the certified values exceed 7.3 and $11/2$ by less than 10^{-6}), except that of `0015`, which comes from the search, is not certified and is shown for comparison only. Contacts are counted one per pair. †: not in the archive and not used by the proof.

stored lattice of	contacts	degree 2	degree 3	degree 4
`0001`	120	7.3000	–	–
`0014`, `0059`	130	5.5000	$5/2$	–
`0015`	135	8.5	$47/14$	$29/10$ †

the sums being over multisets of contacts. Then $\|\widehat{E}\|_F^2 \le K\, t(E)^2$ *for every* $E \in \mathcal{K}(G)$.

Proof. On the cone every $\ell_c(E)$ is nonnegative, so the right sides are nonnegative, and $t = t(E) \ge 0$ by Lemma 9 a. If $t > 0$, divide (5) by t, or (6) by t^2. Let now $t = 0$. The matrix G lies in the cone, since $G[c] = 4 > 0$, and $t(G) = n$. For $\varepsilon > 0$ the matrix $E + \varepsilon G$ lies in the cone and has trace $n\varepsilon > 0$, so the case already treated gives $\|\widehat{E} + \varepsilon I\|_F^2 \le K\, n^2\varepsilon^2$. Let $\varepsilon \to 0$: $\widehat{E} = 0$, and the inequality holds. □

The case $t = 0$ needs care: the identity alone says nothing there. It also follows from a certificate of Lemma 11 below with zero residual ($\widehat{E} \succeq -a\,t\,I = 0$ and $\operatorname{tr}\widehat{E} = 0$ give $\widehat{E} = 0$); the argument above needs no such certificate. The checker verifies an identity (5) or (6) exactly. It requires G to be integral, multiplies both sides by a common denominator, and compares their symmetric coefficient tensors modulo primes below 2^{24} whose product exceeds twice an explicit bound of the integer coefficients; two integer tensors below that bound that agree modulo all these primes are equal. The coefficients β are stored for one multiset in each orbit of a group of automorphisms of G; the checker verifies that each stored generator g is an integer matrix with $g^{\mathsf T} G g = G$ that permutes the contacts, and expands the orbits. Soundness does not depend on the group: any $\beta \ge 0$ that satisfies the identity is a certificate. Table 3 gives the constants that are verified.

Lemma 11 (a certificate for the least eigenvalue). *Let* G *and* C_G *be as in Lemma 9, and let* $\|\widehat{E}\|_F^2 \le K\, t(E)^2$ *on* $\mathcal{K}(G)$. *Let* $\tilde a \ge 0$ *and* $w_{cd} \ge 0$ $(c, d \in C_G)$ *be rational, and let*

$$R'(E) = E + \tilde a\, t(E)\, G - \sum_{c,d} w_{cd}\, E[c]\, (Gd)(Gd)^{\mathsf T},$$

a symmetric matrix whose entries $R'(E)_{k\ell} = \sum_{i\le j} R'_{ij,k\ell}\, E_{ij}$ *are linear forms in the coordinates* E_{ij} $(i \le j)$. *Then, for every* $E \in \mathcal{K}(G)$,

$$\lambda_{\min}(\widehat{E}) \;\ge\; -a\, t(E), \qquad a = \tilde a + 2\, g(G^{-1})\, g(G)\, \overline{\sqrt{K}} \sum_{i\le j} \sum_{k\le \ell} \bigl|R'_{ij,k\ell}\bigr|,$$

where $g(\cdot)$ *is the Gershgorin bound of the largest eigenvalue and* $\overline{\sqrt{K}} \ge \sqrt{K}$ *is rational.*

Proof. On the cone, $\sum_{c,d} w_{cd} E[c]\,(Gd)(Gd)^{\mathsf{T}} \succeq 0$, so $E + \tilde{a}\,t\,G \succeq R'(E)$. Every entry of E is at most $g(G)\|\widehat{E}\|_F \le g(G)\sqrt{K}\,t$ in absolute value (proof of Lemma 9 b). Hence the operator norm of $R'(E)$, which is at most the sum of the absolute values of all its entries, is at most $2\,g(G)\sqrt{K}\,t\sum|R'_{ij,k\ell}|$. Multiply $E + \tilde{a}\,t\,G \succeq R'(E)$ on both sides by $G^{-1/2}$: $\widehat{E} + \tilde{a}\,t\,I \succeq -\lambda_{\max}(G^{-1})\,\|R'(E)\|\,I$, and $\lambda_{\max}(G^{-1}) \le g(G^{-1})$. □

An identity with zero residual proves eutaxy. If $R' = 0$, then $E + \tilde{a}\,t(E)\,G = \sum_{c,d} w_{cd} E[c]\,(Gd)(Gd)^{\mathsf{T}}$ for every symmetric E; the trace against G^{-1} gives $(1 + n\tilde{a})\,t(E) = 4\sum_c\big(\sum_d w_{cd}\big)E[c]$, since $G[d] = 4$, that is $G^{-1} = \sum_c \varrho_c\,cc^{\mathsf{T}}$ with $\varrho_c = 4\sum_d w_{cd}/(1 + n\tilde{a}) \ge 0$. Algorithm 7 tests this relation with these coefficients.

The checker accepts K by Lemma 9 b (the certificate stores $\tilde{K}$ and the $\alpha_{cc'}$; R is recomputed exactly) or by Lemma 10. It takes a and b from c, with s an upper bound of the square root, and replaces a by the number of Lemma 11 when a certificate $(\tilde{a}, w)$ is stored and gives a smaller value; the residual coefficients $R'_{ij,k\ell}$ are recomputed exactly, and they vanish when the certificate is an exact identity. These constants hold on the whole cone. (Constants computed only on the ten directions in which a chain matrix $G(\tau, u)$ can move are not valid for an admissibly marked Q, and the checker does not use them.) We write K, a, b for the constants of $\mathcal{K}(G_c)$ ($n = 10$) and K_P, a_P, b_P for those of $\mathcal{K}(P)$ ($n = 9$); every one of the ten forms P is eutactic.

Lemma 12 (a quadratic minorant; [6, Lemma 37]). *Let $\widehat{E}$ be symmetric of size n with $t = \operatorname{tr}\widehat{E} > 0$, $\|\widehat{E}\|_F^2 \le Kt^2$ and $\widehat{E} \succeq -at\,I$, where $a > 0$, $K \ge 1/n$ and $at < 1$. With $b_m = \frac{K+a}{1+na}$, $A = \frac{nb_m - 1}{a + b_m}$ and $B = \frac{na+1}{a+b_m}$,*

$$\log\det(I + \widehat{E}) \;\ge\; m(t) := A\log(1 - at) + B\log(1 + b_m t).$$

The function m is concave, and, if $K > 1/n$, increasing on $[0, t_m]$, $t_m = 1/(nab_m)$, where $at_m < 1$.

Proof. The inequality is [6, Lemma 37], where it is proved. The function m is a positive combination of concave functions; $m'(t) = -aA/(1 - at) + b_m B/(1 + b_m t)$ vanishes only at $t = (b_m B - aA)/(ab_m(A + B)) = 1/(nab_m)$, and, for $K > 1/n$, $at_m = 1/(nb_m) < 1$ because $nb_m > 1$ (for $K = 1/n$, $A = 0$ and m increases on $[0, \infty)$). □

We write m, t_m for the minorant of $\mathcal{K}(G_c)$ ($n = 10$, constants K, a) and m_P, t_P for that of $\mathcal{K}(P)$ ($n = 9$, constants K_P, a_P). The checker evaluates them with rational bounds of the logarithms rounded in the direction that weakens the bound, and compares $d_c\,e^m$ with d_{cut} through a rational lower bound of the exponential.

Lemma 13 (marked radial barrier; [6, Lemma 36]). *Let $I + \widehat{E} \succ 0$ have size n and let $f(s) = \log\det(I + s\widehat{E})$ for $0 \le s \le 1$. Then f is concave and $f'(1) = n - \operatorname{tr}\big((I + \widehat{E})^{-1}\big)$; hence $f(s) \le f(1) + (1 - s)\max\big(0, \operatorname{tr}((I + \widehat{E})^{-1}) - n\big)$, and f is nondecreasing on $[0, 1]$ if $\operatorname{tr}\big((I + \widehat{E})^{-1}\big) \le n$.*

The proof is in [6, Lemma 36].

Lemma 14 (the rank-9 member). *For every Q admissibly marked by P, with S and s_N as in Lemma 6: $s_N \ge 0$, $\lambda_{\max}(P^{-1}Q) \le 1 + b_P s_N$, and*

$$\log\det S \;\ge\; m_P\big(\min(s_N, t_P)\big).$$

Proof. $Q-P \in \mathcal{K}(P)$, and its matrix $\widehat{E}$ relative to P is $S-I$; Lemma 9 gives $s_N = t(Q-P) \ge 0$ and the spectrum of $S-I$ in $[-a_P s_N, b_P s_N]$. By **(Mk)**, $\operatorname{tr}(S^{-1}) = \operatorname{tr}(XP) \le 9$, so $f(\sigma) = \log\det(I + \sigma(S-I))$ is nondecreasing on $[0,1]$ (Lemma 13). For $s_N > 0$ and $\sigma = \min(1, t_P/s_N)$, the matrix $\sigma(S-I)$ lies in the cone and has trace $\sigma s_N = \min(s_N, t_P)$, so Lemma 12 gives $\log\det S = f(1) \ge f(\sigma) \ge m_P(\min(s_N, t_P))$. For $s_N = 0$, $S = I$. □

Lemma 15 (a bound on u). *Let $z_j \in \mathbb{Z}^9 \setminus \{0\}$ and rationals $q_j \ge 0$ be given (a frame); let $W_c = \sum_j q_j (c \cdot z_j)^2$ for $c \in C$, let $S_9(W)$ be the sum of the nine largest W_c, and, for a centre τ_c and $r \ge 0$, let $\varphi_j = \min\big(\frac12, \|z_j \cdot \tau_c\| + \sqrt{P^{-1}[z_j]}\, r\big)$. Every chain over P with $P[\tau - \tau_c - k] \le r^2$ for some $k \in \mathbb{Z}^9$ has*

$$u \;\le\; U(r) := \min\Big(\frac{S_9(W)}{4\sum_j q_j(1-\varphi_j^2)},\; \sqrt[5]{4/3}\Big)$$

if the denominator is positive; otherwise, and for an empty frame, $U(r) = \sqrt[5]{4/3}$.

Proof. $|z_j \cdot (\tau - \tau_c - k)| \le \sqrt{P^{-1}[z_j]\, P[\tau - \tau_c - k]}$, so $\|z_j \cdot \tau\| \le \varphi_j$, and **(L2)** gives $X[z_j] \ge u(1-\varphi_j^2)$. By (1), $\sum_j q_j X[z_j] = \sum_c \lambda_c W_c \le S_9(W)/4$: at most nine weights λ_c reach $\frac14$, and their sum is at most $\frac94$. Add up, and use **(u)**. □

If $\sum_j q_j z_j z_j^{\mathsf T} = P$, then $W_c = P[c] = 4$ and $S_9(W)/4 = 9$; if moreover the z_j are minimal vectors of the dual with $P^{-1}[z_j] = \frac34$, then $\sum_j q_j = 12$, and since $\varphi_j \le \frac12$ always, $U(r) \le 1$ for every centre and every radius. Such *twelve-weight frames* exist for `0001`, `0014`, `0059` and `0015`, whose stored lattices have $u_0 = 1$. The checker does not test the identity $\sum_j q_j z_j z_j^{\mathsf T} = P$ and does not need it: by Lemma 15, every frame of nonzero vectors of $\mathbb{Z}^9$ with nonnegative weights gives a valid bound, and the checker computes $U(r)$ from the frame as it is stored. For the stored frames of these forms it finds $U(r) = 1$.

Lemma 16 (localisation). *Let (τ_c, u_c) be a centre, $r > 0$ rational, $U \ge U(r)$ (Lemma 15), $v_{\mathrm{lo}} = u_c/U$ and $v_{\mathrm{hi}} = \overline{d}_{\mathrm{cut}}/d_c$. Let a chain over P satisfy $P[\tau - \tau_c] \le r^2$ and $d < d_{\mathrm{cut}}$, and let $t = t(G(Q,\tau,u) - G_c)$. Then $v \ge v_{\mathrm{lo}}$ and $\log\det S + \log v < \log v_{\mathrm{hi}}$. Moreover:*

a *if $s_1 \in (0, t_P]$ satisfies $m_P(s_1) \ge \log(v_{\mathrm{hi}}/v_{\mathrm{lo}})$, then $s_N < s_1$ and*

$$t \;<\; T_b := u_c r^2 - 1 + \max\big(v_{\mathrm{hi}},\; s_1(1 + u_c b_P r^2) + v_{\mathrm{lo}}\big);$$

b *if (τ_c, u_c) is a cone centre and $t \le t_m$, then $t < t_L$ for every $t_L \in (0, t_m]$ with $d_c\, e^{m(t_L)} \ge d_{\mathrm{cut}}$;*

c *if (τ_c, u_c) is a cone centre, $\varepsilon = Ur^2 + U/u_c - 1$, $\varepsilon_+ = \max(\varepsilon, 0)$, and $d_c\, e^{m(t_1) - \varepsilon_+} \ge d_{\mathrm{cut}}$ for some $t_1 \in (0, t_m]$, then $t < t_1$.*

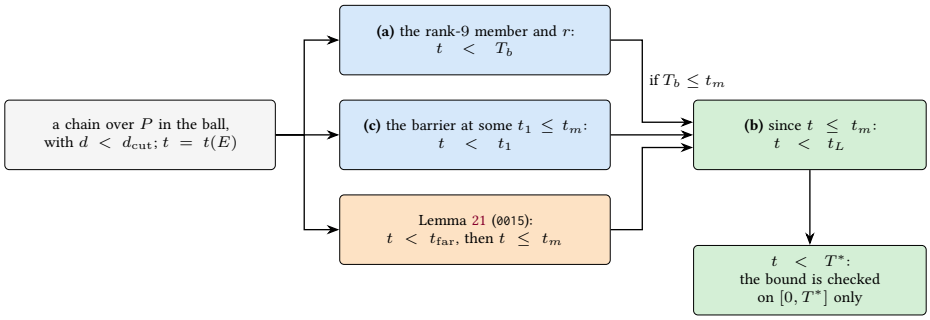


Fig. 7. How a ball bounds the trace t of a chain (Lemmata 16 and 21). Item (a) bounds t by T_b directly; when $t \le t_m$ is known, by (a), by the barrier (c), or by Lemma 21 for 0015, item (b) gives the sharper bound t_L, which does not depend on the radius. The balls of 0001 and 0014 have $T^* = t_L$; that of 0059 uses (c).

Proof. $v = u_c/u \ge u_c/U$ by Lemma 15, and $\log\det S + \log v = \log(d/d_c) < \log(d_{\mathrm{cut}}/d_c) \le \log v_{\mathrm{hi}}$ by Lemma 6 a. a: if $s_N \ge s_1$, Lemma 14 gives $\log\det S \ge \min(m_P(s_1), m_P(t_P)) = m_P(s_1)$, since m_P increases on $[0, t_P]$; then $\log v < \log v_{\mathrm{hi}} - m_P(s_1) \le \log v_{\mathrm{lo}}$, a contradiction. So $s_N < s_1 \le t_P$, and $v < v_{\mathrm{hi}} e^{-m_P(s_N)}$. By Lemma 6 b and Lemma 14, $Q[\delta] \le \lambda_{\max}(P^{-1}Q)\, P[\delta] \le (1 + b_P s_N) r^2$, so $t < \phi(s_N) + u_c r^2 - 1$ with $\phi(s) = s(1 + u_c b_P r^2) + v_{\mathrm{hi}} e^{-m_P(s)}$. The function e^{-m_P} is log-convex, hence convex, so ϕ is convex on $[0, s_1]$ and is at most $\max(\phi(0), \phi(s_1)) \le \max\big(v_{\mathrm{hi}}, s_1(1 + u_c b_P r^2) + v_{\mathrm{lo}}\big)$, because $v_{\mathrm{hi}} e^{-m_P(s_1)} \le v_{\mathrm{lo}}$. b: $E = G(Q, \tau, u) - G_c \in \mathcal{K}(G_c)$, $t \ge 0$, and $at < 1$; if $t > 0$, Lemmata 12 and 6 a give $m(t) \le \log(d/d_c) < \log(d_{\mathrm{cut}}/d_c) \le m(t_L)$, and m increases on $[0, t_m]$. c: suppose $t \ge t_1$ and let $s = t_1/t \in (0, 1]$. The matrix $s\widehat{E}$ is in the cone with trace t_1, so Lemma 12 gives $\log\det(I + s\widehat{E}) \ge m(t_1)$. By Lemma 13 and Lemma 6 c, with **(Mk)**, $u \le U$ and $P[\delta] \le r^2$, $\log\det(I + s\widehat{E}) \le \log(d/d_c) + (1 - s)\varepsilon_+ < \log(d_{\mathrm{cut}}/d_c) + \varepsilon_+$. Together, $d_c e^{m(t_1) - \varepsilon_+} < d_{\mathrm{cut}}$, a contradiction. □

Figure 7 shows how the three items bound the trace of a chain.

The covering bound. Let (τ_c, u_c) be a cone centre and M_c the lattice of G_c (for the stored gluing, $M_c = M_0$). Let the Delaunay cells D of M_c (for the metric G_c) have centres p_D, vertex sets V_D and squared radii R_D^2. A *vertex constant* of D is a rational μ_D with $E[v - p_D] \le \mu_D t(E)$ for every $v \in V_D$ and every $E \in \mathcal{K}(G_c)$ with $\|\widehat{E}\|_F^2 \le K t(E)^2$. The checker accepts $\mu_D = b\, R_D^2$ (since $E[w] \le \lambda_{\max}(\widehat{E}) G_c[w]$), or a certificate: for each vertex, rationals $\tilde{\mu}$ and $q_c \ge 0$ with the exact residual

$$R_m = \tilde{\mu} G_c^{-1} - w w^{\mathsf{T}} - \sum_c q_c\, c c^{\mathsf{T}}, \qquad w = v - p_D,$$

and

$$\mu_D \ge \tilde{\mu} + \sqrt{K \operatorname{tr}\big((G_c R_m)^2\big)} \quad \text{(the square root rounded up).}$$

Indeed $E[w] = \tilde{\mu}\, t - \sum_c q_c E[c] - \operatorname{tr}(E R_m) \le \tilde{\mu}\, t + \|\widehat{E}\|_F \, \|G_c^{1/2} R_m G_c^{1/2}\|_F$. A cell D is *deep* when its data contain a distribution π on V_D with

$$\pi \ge 0, \qquad \sum_v \pi_v = 1, \qquad \pi_v = \pi_{2p_D - v},$$
$$\sum_v \pi_v (v - p_D) = 0, \qquad \sum_v \pi_v (v - p_D)(v - p_D)^{\mathsf{T}} = \kappa_D G_c^{-1},$$

$\kappa_D = R_D^2/10$, a *central radius* $c_D > 0$ (every point y with $G_c[y - p_D] \le c_D^2$ is the mean of a distribution on V_D with second moment $\kappa_D G_c^{-1}$ about p_D; Lemma 18) and an *inradius* $R_{0,D} > 0$ (the ball of G_c-radius $R_{0,D}$ about p_D lies in D). For a deep cell let $\tilde{c}_D = \min(c_D, R_{0,D}/2)$ and

$$t_D = \frac{4\tilde{c}_D (R_{0,D} - \tilde{c}_D)}{(\mu_D - \kappa_D) + 4\tilde{c}_D (R_{0,D} - \tilde{c}_D)\, a} \qquad (t_D = +\infty \text{ if } \mu_D \le \kappa_D).$$

Define

$$\mathcal{R}(t) = \max_D h_D(t), \qquad h_D(t) = \begin{cases} \kappa_D (10 + t), & D \text{ deep and } t \le t_D, \\ R_D^2 + \mu_D t, & \text{otherwise.} \end{cases} \tag{7}$$

Lemma 17 (covering bound). *For every $E \in \mathcal{K}(G_c)$ with $\|\widehat{E}\|_F^2 \le K t^2$, $\lambda_{\min}(\widehat{E}) \ge -at$ and $at < 1$, where $t = t(E)$: $\rho^2_{G_c + E}(\mathbb{Z}^{10}) \le \mathcal{R}(t)$.*

Proof. Let $G = G_c + E$ and let $y \in \mathbb{R}^{10}$; it lies in a cell D of the Delaunay tiling of M_c. For every vertex, $G[v - p_D] = R_D^2 + E[v - p_D] \le R_D^2 + \mu_D t$, so the proof of Lemma 5 gives $\min_v G[v - y] \le R_D^2 + \mu_D t$: this is the second line of (7). Let now D be deep and $t \le t_D$; the mixture argument of [7, Lemma 14] (one mixture bound for deep cells) applies in the metric G_c, with the central radius $\tilde{c}_D$ in place of its κ; we repeat the argument. Write $B = (\mu_D - \kappa_D)t$, so that $G[v - p_D] \le \kappa_D \operatorname{tr}(G_c^{-1} G) + B$ with $\operatorname{tr}(G_c^{-1} G) = 10 + t$, and let $\lambda = \lambda_{\min}(G_c^{-1/2} G G_c^{-1/2}) \ge 1 - at > 0$. If $G_c[y - p_D] \le \tilde{c}_D^2$, y is the mean of a distribution of second moment $\kappa_D G_c^{-1}$ about p_D, and Lemma 4 gives $\min_v G[v - y] \le \kappa_D (10 + t) - G[y - p_D] \le \kappa_D (10 + t)$. If y is farther from p_D, at G_c-distance ω, mix a central distribution of mean $p_D + \tilde{c}_D (y - p_D)/\omega$ with a distribution on the boundary of D of mean on the same ray, at G_c-distance at least $R_{0,D}$. (The point p_D is interior to D, since $R_{0,D} > 0$, so the ray from p_D through y leaves D at a point of its boundary, which is a convex combination of vertices of D.) The mixture has mean y and weight θ on the boundary distribution, with $0 \le \theta \le 1$ because $y \in D$ and $\theta \le (\omega - \tilde{c}_D)/(R_{0,D} - \tilde{c}_D)$; its mean squared G-distance is at most $\kappa_D (10 + t) + \theta B - \lambda \omega^2$. If $B \le 0$, that is $\mu_D \le \kappa_D$, then $\theta B \le 0$ and we are done. Otherwise $\theta B \le B(\omega - \tilde{c}_D)/(R_{0,D} - \tilde{c}_D)$; since $\omega^2 \ge 4\tilde{c}_D (\omega - \tilde{c}_D)$ and $B \le 4\tilde{c}_D (R_{0,D} - \tilde{c}_D)(1 - at)$ for $t \le t_D$, the last two terms add up to at most 0 (Figure 8). In all cases $\min_v G[v - y] \le h_D(t) \le \mathcal{R}(t)$, and y was arbitrary. □

Lemma 18 (central radii). *Let π be a distribution on the vertex set V_D of a cell with centre p_D, with $\pi_v = \pi_{2p_D - v}$, mean p_D and second moment $\kappa_D G_c^{-1}$ about p_D; write $w_v = v - p_D$, so that $G_c[w_v] = R_D^2$.*

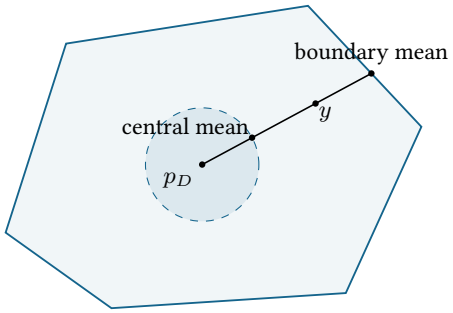


Fig. 8. The averaging argument on a fixed reference cell of the stored lattice, used for the deep cells in Lemma 17: mix two distributions on the vertices, a central one (dashed disc of radius $\tilde{c}_D$) and one on the boundary of the cell; their mixture has mean y. A planar illustration (reused from [7], with the labels of this paper).

a $c_D = \kappa_D / R_D$ *is a central radius.*
b *If* $\sum_v \pi_v (\zeta \cdot w_v)^4 \le m_4 \, (G_c^{-1}[\zeta])^2$ *for every* $\zeta \in \mathbb{R}^{10}$, *then* $c_D = \kappa_D \sqrt{\kappa_D} / \sqrt{m_4}$ *is a central radius.*

Proof. Let $\epsilon : V_D \to [-1, 1]$ be odd, $\epsilon_{2p_D - v} = -\epsilon_v$, and let $\pi'_v = \pi_v (1 + \epsilon_v) \ge 0$. Since π is even, $\sum_v \pi_v \epsilon_v = 0$ and $\sum_v \pi_v \epsilon_v w_v w_v^{\mathsf{T}} = 0$: the weights π' form a distribution with the same second moment $\kappa_D G_c^{-1}$ about p_D, and its mean is $p_D + \sum_v \pi_v \epsilon_v w_v$. The set Z of the vectors $\sum_v \pi_v \epsilon_v w_v$, over all odd ϵ, is convex, compact and symmetric, and its support function is $\max_\epsilon \sum_v \pi_v \epsilon_v \, (\zeta \cdot w_v) = \sum_v \pi_v |\zeta \cdot w_v|$ (take ϵ_v equal to the sign of $\zeta \cdot w_v$, which is odd). So a point y is such a mean as soon as

$$\zeta \cdot (y - p_D) \;\le\; \sum_v \pi_v \, |\zeta \cdot w_v| \qquad \text{for every } \zeta.$$

The left side is at most $\sqrt{G_c^{-1}[\zeta]} \, \sqrt{G_c[y - p_D]}$. For the right side let $Y = \zeta \cdot w_v$ under π, so that $\mathbb{E}Y^2 = \kappa_D G_c^{-1}[\zeta]$. a: $|Y| \le R_D \sqrt{G_c^{-1}[\zeta]}$, so $\mathbb{E}|Y| \ge \mathbb{E}Y^2 / \max |Y| \ge (\kappa_D / R_D) \sqrt{G_c^{-1}[\zeta]}$. b: by Hölder's inequality with the exponents $\frac{3}{2}$ and 3, $\mathbb{E}Y^2 = \mathbb{E}\big(|Y|^{2/3} |Y|^{4/3}\big) \le (\mathbb{E}|Y|)^{2/3} (\mathbb{E}Y^4)^{1/3}$, so

$$\mathbb{E}|Y| \;\ge\; \frac{(\mathbb{E}Y^2)^{3/2}}{(\mathbb{E}Y^4)^{1/2}} \;\ge\; \frac{(\kappa_D G_c^{-1}[\zeta])^{3/2}}{m_4^{1/2} \, G_c^{-1}[\zeta]} = \frac{\kappa_D \sqrt{\kappa_D}}{\sqrt{m_4}} \sqrt{G_c^{-1}[\zeta]}.$$

In both cases every y with $G_c[y - p_D] \le c_D^2$ satisfies the condition. □

The checker accepts a central radius $c_D \le \kappa_D / R_D$, or $c_D \le \kappa_D \sqrt{\kappa_D} / \sqrt{m_4}$ when the deep data contain m_4 with an exact sum-of-squares identity, with a positive semidefinite rational Gram matrix, for the fourth-moment bound of b. In both cases it checks that π is even. For the stored balls of `0014` and `0059` the second option is used. The inradius is checked against the exact facets of D.

Theorem 19 (general simple ball). *Let (τ_c, u_c) be a cone centre with constants K, a of $\mathcal{K}(G_c)$, accepted vertex constants μ_D and deep data of the Delaunay cells of M_c, and $\mathcal{R}$ as in (7). Let P be eutactic with constants K_P, a_P, b_P, let $r > 0$ be rational and $U \geq U(r)$ (Lemma 15). Let*

$$T^* = \min\{T_b,\ t_L,\ t_1\},$$

where each term is present only under its condition in Lemma 16:
- *T_b, if a rational s_1 satisfies the condition of* a *;*
- *t_1, if this rational satisfies the condition of* c *;*
- *t_L, any rational with the property of* b*, if T_b is present and $T_b \leq t_m$, or if t_1 is present.*

Suppose that at least one term is present, so that T^* is defined, that $aT^* < 1$, and that at both ends t of every piece of $[0, T^*]$ cut at the points t_D

$$\mathcal{R}(t) < 4, \qquad d_c\, e^{m(t)}\big(4 - \mathcal{R}(t)\big) \ \geq\ 972,$$

where $\mathcal{R}$ is taken with the rows of (7) that are valid on the whole piece. Then (τ_c, r) satisfies (B): every chain over P with $P[\tau - \tau_c] \leq r^2$ has $d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$.

Proof. On a piece, e^m is log-concave (Lemma 12) and $4 - \mathcal{R}$ is a minimum of affine functions, hence concave; it is positive at the ends, hence on the piece, and the product is log-concave there. So the product is at least 972 on the whole of $[0, T^*]$. Let now a chain over P satisfy $P[\tau - \tau_c] \leq r^2$ and $d < d_{\mathrm{cut}}$, and let $E = G(Q, \tau, u) - G_c$ and $t = t(E)$. By **(M)**, $E \in \mathcal{K}(G_c)$, so $t \geq 0$, $\|\widehat{E}\|_F^2 \leq Kt^2$ and $\lambda_{\min}(\widehat{E}) \geq -at$ (Lemma 9). By Lemma 16, $t < T^*$: by a, $t < T_b$; if $T_b \leq t_m$, then $t \leq t_m$ and b gives $t < t_L$; by c, $t < t_1 \leq t_m$, and b gives $t < t_L$. Hence $at < 1$, and Lemmata 6 a and 12 give $d = d_c \det(I + \widehat{E}) \geq d_c\, e^{m(t)}$ (for $t = 0$, $E = 0$ by Lemma 9 b, and $d = d_c$). By Lemma 17, $\rho^2(M) \leq \mathcal{R}(t) < 4$, and by Lemma 3,

$$\Delta \ \geq\ d\,\big(4 - \rho^2(M)\big) \ \geq\ d_c\, e^{m(t)}\big(4 - \mathcal{R}(t)\big) \ \geq\ 972,$$

which is what (B) asks. □

The radius enters the theorem only through the localisation, that is through U, T_b and ε; the bound itself is checked on $[0, T^*]$, and in the typical case $T^* = t_L$, which does not depend on r. Only the minimal vectors of G_c and of P, the marking and the shortest dual vector are used: the determinant $\det Q$ and the shape of Q may vary freely within the conditions of Definition 1.

Definition 20 (simple ball). *The* simple ball *of a form is the ball (τ_0, r) of Theorem 19 at the stored gluing, together with its images $(g\tau_0 + k, r)$, where g runs over the group generated by $-I$ and the automorphisms of P named in the setup of the tree, and k over $\mathbb{Z}^9$ (Lemma 2).*

At $t = 0$ the theorem asks for $d_0(4 - \mathcal{R}(0)) \geq 972$, where $\mathcal{R}(0) \geq \rho^2(M_0)$. The inequality is not strict: a stored lattice without margin, $d_0\big(4 - \rho^2(M_0)\big) = 972$, can pass at $t = 0$. For such a lattice the theorem fails further on (Section 3.4). Figure 10

Table 4. The ball of Theorem 19 at the stored gluing of each of the five forms (checked by Algorithm 6). *Needed*: the range, over 16 random rays from τ_0, of the P-distance at which the pointwise bound of the rank-10 leaves of the pack first reaches d_{cut} (floating point; a diagnostic, not a proof). U/u_0: the bound of Lemma 15 at $r = 0$. †: not in the archive and not used by the proof.

form	needed	r	U/u_0	remark
`0001`	0.032–0.036	0.322485	1	$T^* = t_L$
`0014`	0.088–0.092	0.249731	1	$T^* = t_L$
`0059`	0.080–0.099	0.117202 †	1	by `c`; K of degree 2
`0059`		0.343459	1	by `c`; $K = 5/2$ of degree 3; used by the tree
`0014`		0.469436 †	1	$K = 5/2$ of degree 3; not used by the tree
`0052`	0.013–0.082	none	1.03	M_0 not eutactic; the tree needs no ball
`0015`	0.150–0.167	none	1	no margin; a ball by Section 3.4

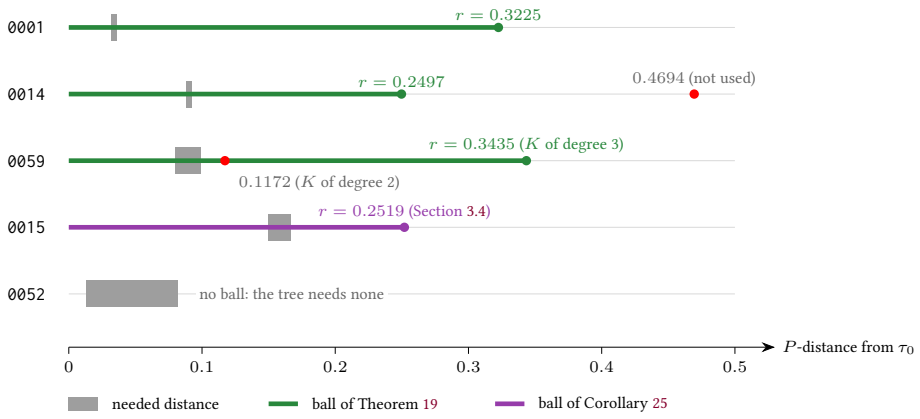


Fig. 9. The certified radius of each ball and the distance that the tree needs (Table 4). The ball must reach beyond the grey range, where the leaves of the reproduction pack begin to close. For `0059` the constant K of degree two gives a ball that barely covers the range, the constant of degree three a comfortable one. For `0015`, Theorem 19 gives no ball; the ball comes from Section 3.4. The red circles mark balls that are not in the archive and are not used by the proof: for `0059` the radius that the constant of degree two would give, for `0014` the radius with the constant of degree three.

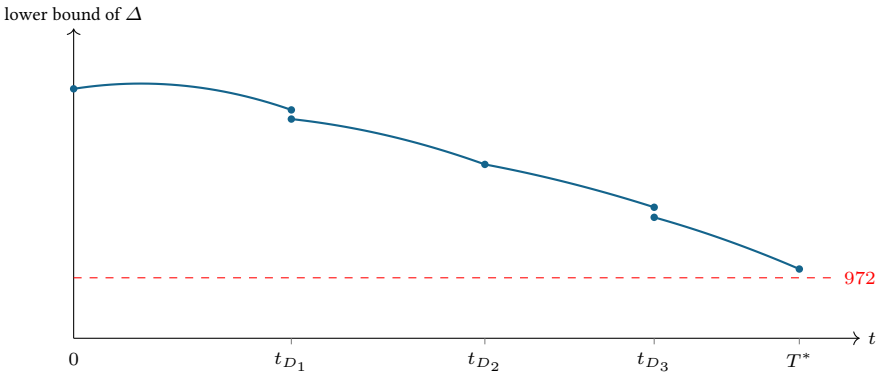


Fig. 10. The bound $d_c\, e^{m(t)}\,(4 - \mathcal{R}(t))$ of Theorem 19, schematically. It is log-concave on each piece of $[0, T^*]$, so it is checked only at the ends of the pieces (dots); it may jump down at a point t_D, where a deep cell switches to its shallow bound. The checker compares each value with 972 in exact arithmetic.

shows the shape of the bound that the checker evaluates, Table 4 the radii, and Figure 9 the radii against the distance that each tree needs.

For `0001`, `0014` and `0059`, whose stored lattices have $u_0 = 1$ and twelve-weight frames, the theorem gives a ball. For `0001` and `0014` the ball covers, with room, the region in which the leaves of the pack cannot work, and the trees of both forms close (Table 7). The radii are $64497/200000$ and $249731/10^6$.

For `0059` the cone of P is too flat for `a` ($m_P(t_P) = 0.048$, below $\log(d_{\text{cut}}/d_0) = 0.083$), and the localisation comes from the barrier `c`: with $U = u_0 = 1, \varepsilon = r^2$, and the radius is, up to the step of the search, the largest for which $m(t_1) - r^2 \geq \log(d_{\text{cut}}/d_0)$ at some $t_1 \leq t_m$. With the constant K of degree two, $m(t_m) = 0.0971$ and $r = 58601/500000$, a margin of 18% over the needed distance. With the constant $K = 5/2$ of degree three (Lemma 10, Table 3) and the certified a, which exceeds $\frac{1}{2}$ by less than $2 \cdot 10^{-9}$, $m(t_m) = 0.2014$ and $r = 343459/10^6$, with $T^* = t_L = 0.094489$. This ball is checked by `l12general2.verify_simple`, which is Algorithm 6 with Lemma 10 for K; the general tree of `0059` uses it, and its exact verification is valid (Table 7). The same constant gives a ball of radius $117359/250000$ for `0014`. That ball, the ball of degree two of `0059` and the constant of degree four of Table 3 are not in the archive and are not used by the proof.

For `0015` the stored lattice has no margin, $d_0(4 - \rho^2(M_0)) = 972$; the theorem accepts $t = 0$ there, but it fails on $(0, T^*]$, and Section 3.4 gives the ball of `0015`. For `0052` the stored lattice is not eutactic, so the theorem does not apply; the tree of `0052` closes with leaves of the pack only.

3.4 Stored lattices without margin

For `0015` the stored lattice has $d_0 = 972$ and $\rho^2(M_0) = 3$, so $d_0(4 - \rho^2(M_0)) = 972$: Lemma 3 has no margin at M_0. Theorem 19 accepts $t = 0$ there, but it gives no ball,

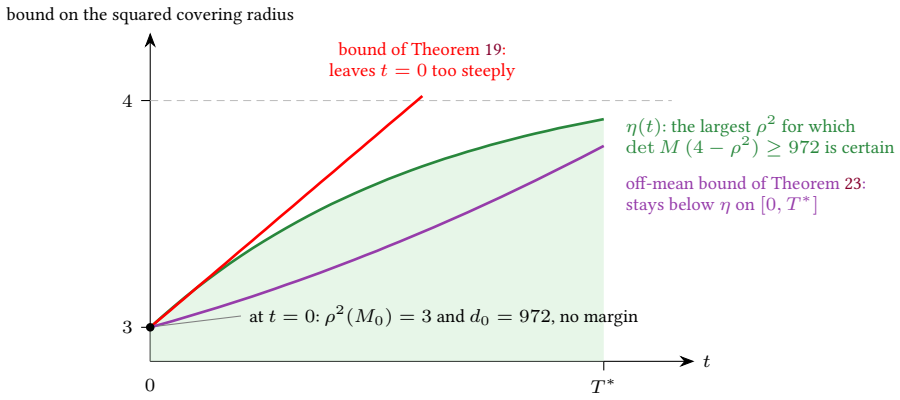


Fig. 11. Why `0015` needs a second theorem (schematic). At the stored lattice the covering bound has no margin: $d_0(4-\rho^2(M_0)) = 972$ exactly. A bound on ρ^2 must stay below $\eta(t)$ on $[0, T^*]$. The bound of Theorem 19 leaves $t = 0$ too steeply; the off-mean bound of Theorem 23 stays below η, which gives the ball of Corollary 25.

for two reasons. The localisation fails: for `0015`, with the constant K of degree two, $m(t_m) = 0.064$ is below $\log(d_{\mathrm{cut}}/972) = 0.1355$, so `c` of Lemma 16 holds at no radius. And the rows of (7) are too weak on $(0, T^*]$. This subsection gives a sharper localisation (Lemma 21) and another covering bound (Theorem 23). Both hold at every cone centre. Figure 11 shows why a second covering bound is needed.

Lemma 21 (sharper localisation). *Let (τ_c, u_c) be a cone centre with constants K, a, minorant m and t_m (Lemma 12). Let $r > 0$ be rational, $U \geq U(r)$, $v_{\mathrm{lo}} = u_c/U$, $v_{\mathrm{hi}} = \overline{d}_{\mathrm{cut}}/d_c$, with $v_{\mathrm{lo}} < v_{\mathrm{hi}}$, and $\varepsilon_+ = \max(Ur^2 + U/u_c - 1, 0)$. Let*

$$h(x) = x\Big(\frac{8x}{9x-1}\Big)^8, \qquad \varphi(x) = \alpha \log x + \beta\Big(\frac{1}{x} - 1\Big) - (x-1), \qquad \alpha = 1 + \beta.$$

Let $\bar\sigma \geq 1$ and $\beta \geq 1$ be rational with $h(\bar\sigma) \geq v_{\mathrm{hi}}/v_{\mathrm{lo}}$, $v_{\mathrm{hi}} \leq \alpha$ and, if $\bar\sigma > \beta$, $\varphi(\bar\sigma) \geq 0$. Let

$$t_{\mathrm{far}} = \alpha \log \frac{v_{\mathrm{hi}}}{v_{\mathrm{lo}}} + v_{\mathrm{lo}} - 1 + u_c\,\bar\sigma\, r^2.$$

Then every chain over P with $P[\tau - \tau_c] \leq r^2$ and $d < d_{\mathrm{cut}}$ has $t < t_{\mathrm{far}}$, where $t = t(G(Q,\tau,u) - G_c)$. If moreover $t_{\mathrm{far}} \leq t_m$, or

$$d_c\, \exp\Big(m(t_m) - \big(1 - \tfrac{t_m}{t_{\mathrm{far}}}\big)\varepsilon_+\Big) \;\geq\; d_{\mathrm{cut}},$$

then $t \leq t_m$, and so $t < t_L$ for every $t_L \in (0, t_m]$ with $d_c\, e^{m(t_L)} \geq d_{\mathrm{cut}}$.

Proof. Let $\sigma_1, \ldots, \sigma_9 > 0$ be the eigenvalues of S. Then $s_N = \sum_i(\sigma_i - 1)$, $\sum_i 1/\sigma_i = \mathrm{tr}(XP) \leq 9$ by **(Mk)**, and $\det S \geq 1$ by Lemma 14. By Lemma 16, $v \geq v_{\mathrm{lo}}$ and $\det S \cdot v < v_{\mathrm{hi}}$; so $v < v_{\mathrm{hi}}$ and $\det S < v_{\mathrm{hi}}/v \leq v_{\mathrm{hi}}/v_{\mathrm{lo}}$.

The largest eigenvalue. Fix i. The eight other eigenvalues have $\sum_{j\neq i} 1/\sigma_j \le 9 - 1/\sigma_i$, so by the inequality between the arithmetic and the geometric mean $\prod_{j\neq i}\sigma_j \ge \big(8/(9-1/\sigma_i)\big)^8$, and $\det S \ge h(\sigma_i)$. Since $h'(x)/h(x) = 9(x-1)/(x(9x-1))$, h increases on $[1,\infty)$. If $\sigma_i \ge \bar\sigma$, then $\det S \ge h(\bar\sigma) \ge v_{\mathrm{hi}}/v_{\mathrm{lo}}$, which is false. So every $\sigma_i < \bar\sigma$.

The trace. $\varphi(1) = 0$ and $\varphi'(x) = -(x-1)(x-\beta)/x^2$, so $\varphi \ge 0$ on $(0,\beta]$ and φ decreases beyond β; with $\varphi(\bar\sigma) \ge 0$, $\varphi \ge 0$ on $(0,\bar\sigma]$. Add up $\varphi(\sigma_i) \ge 0$:

$$s_N \;\le\; \alpha \log\det S + \beta\Big(\sum_i \frac{1}{\sigma_i} - 9\Big) \;\le\; \alpha \log\det S \;<\; \alpha \log \frac{v_{\mathrm{hi}}}{v}.$$

The bound t_{far}. By Lemma 6 b, $t = s_N + u_c Q[\delta] + v - 1$, and $Q[\delta] \le \lambda_{\max}(S)\, P[\delta] \le \bar\sigma r^2$. So $t < \alpha \log(v_{\mathrm{hi}}/v) + v - 1 + u_c\bar\sigma r^2$. The function $v \mapsto v - \alpha\log v$ decreases on $(0,\alpha]$, and $v_{\mathrm{lo}} \le v < v_{\mathrm{hi}} \le \alpha$; so $t < t_{\mathrm{far}}$.

The barrier. If $t_{\mathrm{far}} \le t_m$ we are done. Otherwise suppose $t > t_m$ and let $s = t_m/t$. As in the proof of Lemma 16 c, $m(t_m) \le \log\det(I + s\widehat{E}) \le \log(d/d_c) + (1-s)\varepsilon_+$, and $1 - s < 1 - t_m/t_{\mathrm{far}}$. So $m(t_m) < \log(d_{\mathrm{cut}}/d_c) + (1 - t_m/t_{\mathrm{far}})\varepsilon_+$, against the hypothesis. Hence $t \le t_m$, and Lemma 16 b gives $t < t_L$. □

Item c of Lemma 16 is the case $t_{\mathrm{far}} = \infty$: it takes the worst value $1 - s = 1$, while here t is bounded in the ball, by the marking. The programs use five labels: (C1) is the conclusion $s_N < s_1$ and (C2) the bound T_b, both of item a of Lemma 16; (C3) is its item b and (C4) its item c; (C5) is this lemma; the function `l12gsharp.localise_far` checks its hypotheses exactly, with $\bar\sigma$ and β proposed in floating point.

Lemma 22 (off-mean averaging). *Let (τ_c, u_c) be a cone centre with constant K. Let $y \in \mathbb{R}^{10}$, and let π be a probability distribution on finitely many points v of $\mathbb{Z}^{10}$; its mean need not be y. Let*

$$\vartheta_\pi = \sum_v \pi_v\, G_c[v-y], \qquad \Sigma_\pi = \sum_v \pi_v\,(v-y)(v-y)^{\mathsf T}, \qquad \widehat{\Sigma}_\pi = G_c^{1/2}\,\Sigma_\pi\,G_c^{1/2}.$$

Then, for every $E \in \mathcal{K}(G_c)$ and $t = t(E)$,

$$\min_v\,(G_c + E)[v-y] \;\le\; B_\pi(t) := \vartheta_\pi\Big(1 + \frac{t}{10}\Big) + t\sqrt{K}\,\Big\|\widehat{\Sigma}_\pi - \frac{\vartheta_\pi}{10} I\Big\|_F.$$

Proof. A minimum is at most a mean: $\min_v (G_c+E)[v-y] \le \vartheta_\pi + \operatorname{tr}(E\Sigma_\pi)$, and $\operatorname{tr}(E\Sigma_\pi) = \operatorname{tr}(\widehat{E}\widehat{\Sigma}_\pi)$. Here $\operatorname{tr}\widehat{\Sigma}_\pi = \vartheta_\pi$ and $\operatorname{tr}\widehat{E} = t$, so $\operatorname{tr}(\widehat{E}\widehat{\Sigma}_\pi) = \frac{\vartheta_\pi}{10}t + \operatorname{tr}\big(\widehat{E}(\widehat{\Sigma}_\pi - \frac{\vartheta_\pi}{10}I)\big)$. The Cauchy–Schwarz inequality and $\|\widehat{E}\|_F \le \sqrt{K}\,t$ (Lemma 9) bound the last term. □

Theorem 23 (off-mean covering bound). *Let (τ_c, u_c) be a cone centre with constants K, a, b of $\mathcal{K}(G_c)$ and minorant m, with $d_c\big(4 - \rho^2(M_c)\big) \ge 972$. Let*

$$\eta(t) = 4 - \frac{972}{d_c\, e^{m(t)}}.$$

Let $T > 0$ be rational with $aT < 1$. Suppose that for every Delaunay cell D of M_c, with centre p_D and squared radius R_D^2, and for every point y of D, with $\omega^2 = G_c[y - p_D]$, one of the following holds:

i $(1+bT)\,(R_D^2-\omega^2)\le\eta(T)$;
ii *there is a distribution π as in Lemma 22 with $\vartheta_\pi\le\eta(0)$ and $B_\pi(T)\le\eta(T)$.*
Then every $E\in\mathcal{K}(G_c)$ with $t=t(E)\le T$ satisfies

$$\rho^2_{G_c+E}(\mathbb{Z}^{10})\;\le\;\eta(t),\qquad\textit{hence}\qquad\det(G_c+E)\,\big(4-\rho^2_{G_c+E}(\mathbb{Z}^{10})\big)\;\ge\;972.$$

Proof. The function m is concave (Lemma 12), so e^{-m} is convex and η is concave on $[0,T]$. Let $y\in\mathbb{R}^{10}$; it lies in a cell D. In case ii, B_π is affine in t, at most η at 0 and at T, hence at most η on $[0,T]$, and Lemma 22 gives $\min_v(G_c+E)[v-y]\le\eta(t)$. In case i, some vertex v of D has $G_c[v-y]\le R_D^2-\omega^2$ (Lemma 4, with weights that have mean y), and $(G_c+E)[v-y]\le(1+bt)(R_D^2-\omega^2)$ by Lemma 9 c. This bound is affine in t; at $t=0$ it is at most $R_D^2\le\rho^2(M_c)\le\eta(0)$, and at T it is at most $\eta(T)$. So $\rho^2_{G_c+E}(\mathbb{Z}^{10})\le\eta(t)$. Finally $\det(G_c+E)\ge d_c\,e^{m(t)}$ (Lemma 12; for $t=0$, $E=0$), and $4-\eta(t)=972\,e^{-m(t)}/d_c$. □

The theorem asks for nothing at $t=0$ beyond $d_c(4-\rho^2(M_c))\ge 972$: a centre without margin is allowed. It uses no vertex constant and no linear program; the only inputs are K, a and the Delaunay cells.

The distributions. The checker finds the distribution of ii in closed form. Let D be a cell with centre p, vertex set V and $w_v=v-p$, so that $G_c[w_v]=R^2$; let $\kappa=R^2/10$ and let λ be the uniform distribution on V. Write $y=p+\omega\,e$ with $G_c[e]=1$, choose $\omega_1=\min(\omega,q)$ for a number $q\ge0$ given below, and let $x_1=\omega_1 e$ and $\beta_1=\omega^2-2\omega\omega_1$. For a distribution π on V with mean $p+x_1$ and $W_\pi=\sum_v\pi_v w_v w_v^{\mathsf T}$, a direct computation gives

$$\vartheta_\pi=R^2+\beta_1,\qquad\Sigma_\pi=W_\pi+\beta_1\,ee^{\mathsf T},\qquad\widehat{\Sigma}_\pi-\frac{\vartheta_\pi}{10}I=\big(\widehat{W}_\pi-\kappa I\big)+\beta_1\big(\hat e\hat e^{\mathsf T}-\tfrac{1}{10}I\big),\tag{8}$$

with $\hat e=G_c^{1/2}e$, and the last matrix has Frobenius norm $\sqrt{9/10}\,|\beta_1|$.

Lemma 24 (three distributions). *With the notation above, suppose that λ has mean p.*

N *No tilt. Take $q=0$ and $\pi=\lambda$. Then $\vartheta_\pi=R^2+\omega^2$ and $\|\widehat{\Sigma}_\pi-\frac{\vartheta_\pi}{10}I\|_F\le\|\widehat{W}_\lambda-\kappa I\|_F+\sqrt{9/10}\,\omega^2$.*

L *Linear tilt. Let W_λ be invertible, let $s_v=w_v^{\mathsf T}W_\lambda^{-1}x_1$, let $q\le1/\max_v g_v$ with $g_v^2=G_c^{-1}[W_\lambda^{-1}w_v]$, and let $\ell\ge0$ satisfy $\ell^2G_c-W_\lambda^{-1}Q_3W_\lambda^{-1}\succeq0$, where $Q_3=\sum_{v,v'}\lambda_v\lambda_{v'}(w_v^{\mathsf T}G_cw_{v'})^2\,w_vw_{v'}^{\mathsf T}$. Then $\pi_v=\lambda_v(1+s_v)$ is a distribution with mean $p+x_1$, and $\|\widehat{\Sigma}_\pi-\frac{\vartheta_\pi}{10}I\|_F\le\|\widehat{W}_\lambda-\kappa I\|_F+\ell\omega_1+\sqrt{9/10}\,|\beta_1|$. When $W_\lambda=\kappa G_c^{-1}$, the first term is 0 and $q\le\kappa/R$ is allowed.*

Z *Even tilt. Let $W_\lambda=\kappa G_c^{-1}$, let V be symmetric about p ($2p-v\in V$), let $s_v=w_v^{\mathsf T}G_cx_1$, let $q\le\sqrt{\kappa}$, and let θ_2 satisfy $r^{\mathsf T}\Gamma r\le\theta_2|V|\,r^{\mathsf T}r$ for every $r\in\mathbb{R}^V$ with $\sum_v r_v=0$, where $\Gamma_{vv'}=(w_v^{\mathsf T}G_cw_{v'})^2$. Let $N_1=\sum_v\lambda_v|s_v|$. Then*

$$\pi_v=\lambda_v\Big(1-\frac{N_1}{\kappa}+\frac{2}{\kappa}\max(s_v,0)\Big)$$

is a distribution with mean $p+x_1$, and, when $\omega\le2q$,

$$\Big\|\widehat{\Sigma}_\pi-\frac{\vartheta_\pi}{10}I\Big\|_F^2\;\le\;\frac{9\,\theta_2}{R^2}\,\omega_1^2+\frac{9}{10}\,\beta_1^2.$$

Proof.

N: the mean of λ is p, so $x_1 = 0$, $\beta_1 = \omega^2$, and (8) with the triangle inequality.

L: by the Cauchy–Schwarz inequality $|s_v| \le g_v\omega_1 \le 1$, so $\pi \ge 0$; $\sum_v \lambda_v s_v = 0$ and $\sum_v \lambda_v s_v w_v = W_\lambda W_\lambda^{-1} x_1 = x_1$, so π is a distribution with mean $p + x_1$. Its second moment is $W_\pi = W_\lambda + M_3$ with $M_3 = \sum_v \lambda_v s_v w_v w_v^{\mathsf T}$, and $\|\widehat{M_3}\|_F^2 = x_1^{\mathsf T} W_\lambda^{-1} Q_3 W_\lambda^{-1} x_1 \le \ell^2\omega_1^2$. Use (8). When $W_\lambda = \kappa G_c^{-1}$, $g_v = R/\kappa$.

Z: $\sum_v \lambda_v s_v^2 = \kappa\omega_1^2$, so $N_1 \le \sqrt{\kappa}\,\omega_1 \le \kappa$ and $\pi \ge 0$. By symmetry, $\sum_v \lambda_v |s_v| w_v = 0$ and $\sum_v \lambda_v s_v w_v w_v^{\mathsf T} = 0$. With $\max(s,0) = \frac{1}{2}(s + |s|)$ this gives $\sum_v \pi_v = 1$, the mean $p + x_1$, and

$$W_\pi = \kappa G_c^{-1} + \Theta, \qquad \Theta = \frac{1}{\kappa}\big(A_1 - \kappa N_1 G_c^{-1}\big), \qquad A_1 = \sum_v \lambda_v |s_v|\, w_v w_v^{\mathsf T}.$$

Write $|s_v| = N_1 + r_v$, so that $\sum_v r_v = 0$. Since $\Gamma\mathbf{1} = 10\,|V|\kappa^2\,\mathbf{1}$, we get $\|\widehat{A}_1\|_F^2 = 10\kappa^2 N_1^2 + r^{\mathsf T}\Gamma r/|V|^2$ and $\operatorname{tr}\widehat{A}_1 = 10\kappa N_1$, hence

$$\|\widehat{\Theta}\|_F^2 = \frac{r^{\mathsf T}\Gamma r}{|V|^2\kappa^2} \;\le\; \frac{\theta_2}{\kappa^2}\big(\kappa\omega_1^2 - N_1^2\big) \;\le\; \frac{9\,\theta_2}{R^2}\,\omega_1^2,$$

because $\sum_v \lambda_v r_v^2 = \kappa\omega_1^2 - N_1^2$ and $N_1 \ge \sum_v \lambda_v s_v^2 / \max_v |s_v| \ge \kappa\omega_1/R$. The cross term of (8) is $2\beta_1\,\hat e^{\mathsf T}\widehat{\Theta}\hat e$, since $\operatorname{tr}\widehat{\Theta} = 0$. Here $\hat e^{\mathsf T}\widehat{\Theta}\hat e = (m_3/\omega_1^2 - \kappa N_1)/\kappa$ with $m_3 = \sum_v \lambda_v |s_v|^3$, and $m_3 N_1 \ge (\sum_v \lambda_v s_v^2)^2 = \kappa^2\omega_1^4 \ge \kappa N_1^2\omega_1^2$ by the Cauchy–Schwarz inequality, so it is nonnegative; and $\beta_1 \le 0$ for $\omega \le 2q$. So the cross term is at most 0. □

The cell test of the checker (`112gsharp.check_cover`) is this. A cell with $(1 + bT)R^2 \le \eta(T)$ passes by i. Otherwise let $\omega_*^2 = R^2 - \eta(T)/(1 + bT)$; the points with $\omega \ge \omega_*$ pass by i, and the interval $[0, \omega_*]$ is cut into 400 pieces. On each piece the checker bounds ϑ_π and the Frobenius norm of Lemma 24 from above by their values at the ends (the bounds are monotone or convex in ω), with square roots rounded up, and tests $\vartheta_\pi \le \eta(0)$ and $B_\pi(T) \le \eta(T)$ with rational lower bounds of η. The constants θ_2 and ℓ are proposed in floating point and accepted by an exact test of positive semidefiniteness.

Corollary 25 (a ball at a centre without margin). *Let (τ_c, u_c) be a cone centre, $r > 0$ rational and $U \ge U(r)$. Let T^* be a rational with $t < T^*$ for every chain over P with $P[\tau - \tau_c] \le r^2$ and $d < d_{\mathrm{cut}}$ (Lemma 16 or Lemma 21). If the hypotheses of Theorem 23 hold with $T = T^*$, then (τ_c, r) satisfies (B).*

Proof. For such a chain, $E = G(Q, \tau, u) - G_c \in \mathcal{K}(G_c)$ by **(M)** and $t(E) < T^*$. Theorem 23 and Lemma 3 give $\Delta \ge d\,(4 - \rho^2(M)) \ge 972$. □

The state for `0015`. The stored lattice of `0015` has $u_0 = 1$, a twelve-weight frame ($U = 1$), 135 pairs of contacts and 7302 Delaunay cells up to translations and sign, of squared radii 3, $\frac{8}{3}$, $\frac{7}{3}$, $\frac{9}{4}$ and $\frac{20}{9}$. Its cone has $K = 47/14$ by an identity (5) and $a = \frac{1}{2}$ by an exact identity of Lemma 11. With these constants, c of Lemma 16 holds at $r = 17/125$ and fails at $r = 0.137$, while Lemma 21 holds at $r = 251891/10^6$ (found

by bisection; every step exact), with $T^* = t_L = 0.202775$ and $t_{\mathrm{far}} = 0.4399$. At this T^* the test of Theorem 23 passes: the 3 cells of squared radius 3 by `Z`, the 36 cells of squared radius $\frac{8}{3}$ and the 135 cells of squared radius $\frac{7}{3}$ by `L`, the 7128 others by `i`. The certificate built for the radius $r = 251891/10^6$ is checked by the command `verify` of `112gsharp.py`: it binds P, τ_0, u_0 to the pack, re-checks the two identities, the frame, Lemma 21 and the test of Theorem 23. The full run of this command accepted the certificate: the exact re-check of the 7302 Delaunay cells (Algorithm 5; 317703 facets, no error) and every other step passed. So the ball $(\tau_0, 251891/10^6)$ of `0015` is exactly verified. The tree of `0015` registers this ball and its images (Algorithm 4); its tree has 1463716 leaves, of which 523 of kind `three`, and its exact verification in general mode is valid (Table 7).

3.5 The three kinds of certificate

Definition 26 (kinds of certificate). *A* pure-tree certificate *is a certificate whose tree has leaves of the reproduction pack only. A* simple-ball certificate *is a certificate whose basin leaves rely on the simple ball of the form (Definition 20). A* zero-margin certificate *is a certificate whose basin leaves rely on the ball of Corollary 25 at the stored gluing and on its images.*

The certificate of `0052` is a pure-tree certificate; those of `0001`, `0014` and `0059` are simple-ball certificates; that of `0015` is a zero-margin certificate.

Proposition 27 (what an accepted certificate proves). *If the tree of a certificate is accepted by Algorithm 2 and its ball, if any, is accepted by Algorithm 6 or Algorithm 7, with complete facet lists for its Delaunay cells (Lemma 29), then (E_P) holds for the form.*

Proof. With complete facet lists, the cells are all the Delaunay cells and the inradii are right (Lemma 29). Then Algorithm 6 accepts a simple ball only if the hypotheses of Theorem 19 hold, and Algorithm 7 accepts a ball only if those of Corollary 25 hold; so every registered centre, with its radius, satisfies (B) for every chain over P, and so do its images (Section 3.2). Algorithm 3 accepts a basin leaf only if its box lies in one of these balls. Algorithm 2 checks that the leaves cover the root box and that every leaf of the pack verifies. These are the hypotheses of Lemma 8, which gives (E_P). □

4 How a certificate is checked

4.1 Arithmetic

Every number that enters a decision is a Python integer or a fraction of two Python integers (`fractions.Fraction`); both are exact and unbounded. Determinants are computed by Gaussian elimination over $\mathbb{Q}$, inverses by Gauss–Jordan elimination over $\mathbb{Q}$, positive (semi)definiteness by exact LDL^{T} elimination, and the lattice points in an ellipsoid by Fincke–Pohst enumeration in rational arithmetic [13]. Square roots never appear exactly: a bound that needs $\sqrt{x}$ uses a rational $\underline{s} \le \sqrt{x}$ or $\overline{s} \ge \sqrt{x}$ with absolute

error below 10^{-12}, computed with the integer square root, rounded in the direction that weakens the bound. Facets of a cell come from the double description method of cddlib in rational (GMP) arithmetic [4]; every facet inequality is then re-checked on every vertex of the cell, in integers.

As we already mentioned, floating point only proposes; no acceptance depends on it. An exact routine may start from a floating-point guess, which it then checks exactly. These guesses are: in the flip of a Delaunay cell, the search of the lattice points of a ball, made in floating point with a margin (the result is accepted only if it equals a listed cell, which is itself verified by exact enumeration; Lemma 29); in a basin-leaf test, the integer translation and the choice of the eight nearest centres; the points s_1, t_L, t_1 of Lemma 16 and $\bar{\sigma}$, β of Lemma 21 that a certificate proposes (each is re-checked exactly; a poor one only weakens the bound); the constants θ_2 and ℓ of Lemma 24 (accepted by an exact test of positive semidefiniteness); the frame of Lemma 15 (its weights are only required to be nonnegative); the reduced basis that a lattice reduction proposes (only its unimodularity, checked exactly, matters). One test is made in floating point: the cell volumes of a Delaunay subdivision are added, their sum must be 1, and a check of a ball fails if it is not; this test can reject a correct subdivision but *never accepts a wrong one*. The certificates were found with linear and semidefinite programming in floating point, but the checkers do not use these programs.

One check of this paper uses machine integers. The identities of Lemma 10 are compared modulo primes below 2^{24} with 64-bit integer arrays. Every intermediate value there is a sum of a few hundred products of two residues at most, so it is below 2^{57}, and it is reduced before the next product: no overflow can occur. The bound of the integer coefficients and the choice of the primes are computed with Python integers.

Completeness of the facet lists. The checkers re-check every facet that cddlib lists, but not that none is missing. A separate exact check nails this down (folder `cellcheck` of the archive, run by `verify_all.sh`). For a polytope of dimension k, a non-empty list of genuine facets is complete if and only if every ridge (face of dimension $k-2$) of every listed facet lies in exactly two listed facets: were a facet missing, then, the facet graph being connected, some missing facet would share a ridge with a listed one, and that ridge would lie in one listed facet only. An empty list satisfies the condition and is not complete, so the check also requires at least $k+1$ facets, the least number that a polytope of dimension k can have, at every level. The check applies this criterion recursively (Figure 12), the facets of each face being certified by their own ridges, down to simplices and segments, whose faces are known; cddlib only proposes the candidates, after an exact injective projection of the face on some of its coordinates, and every candidate is re-checked in integers. It also checks that the lists of the checkers are exactly the certified ones. On the four balls it certified 25798 cells and 1516271 facets (4356 cells for `0001`, 7070 for `0014` and `0059`, 7302 for `0015`), in 25 minutes on 16 processes; its last line is `ALL CELL FACET LISTS COMPLETE (25798 cells, 1516271 facets)`.

What is trusted. The list of what the checks rely on, mathematics, software, data and hardware, is the *trust base* of Section 7.

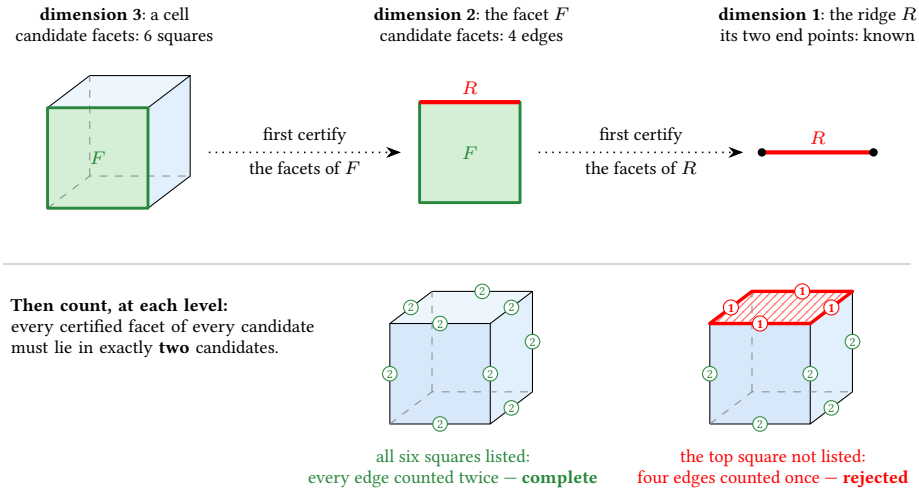


Fig. 12. The check of the facet lists, on a cube. Top: the recursion on the dimension. To certify the candidate facets of a cell, the check first certifies the facets of each candidate F, and for that the facets of each of their candidates R, down to segments and simplices, whose faces are known. Bottom: at each level it then counts. In a complete list every ridge lies in exactly two listed facets; if a facet is missing, its ridges lie in one listed facet only, and the list is rejected. The cells of the paper have dimension 10; the check certified 25798 of them, with 1516271 facets.

4.2 The check of a form

Algorithm 1 (Appendix C) is the whole check of a form; the next subsections give its parts, and Figure 13 shows how they fit. The comparisons of lines 3 and 6 bind every ball to the form: a ball file says which form and which centre it is about, and the checker refuses one that is about anything else. The command `verify CASE --scope general` of the database program runs this algorithm (Sections 5.3 and 6). The records "VALID" stored in the database are not trusted by it: every ball is verified again from its data, and the tree is walked again.

4.3 The tree walk

Algorithm 2 (Appendix C) is the walk of a tree. The tree is checked by the exact verifier of the reproduction pack, the function `verify_box_certificate` of `k11lib.py` [6, Algorithm 8], with one more branch for the basin leaves. The branch is added at run time: the checker reads the source of the function `_box_cert_walk` from the unchanged file `k11lib.py`, inserts the three lines of the branch before the branch of the leaves `three` (it refuses to run if that anchor is missing), compiles the result and installs it in the module. A leaf `"gball"` reaches the last branch of the walk, which calls the function `box_leaf_value` of the pack; a wrapper of that function, installed at run time, answers for the leaves `"gball"` (and only inside a walk, for the form of that walk) by Algorithm 3, and passes every other leaf to the pack's function. Nothing else in the decisions of the

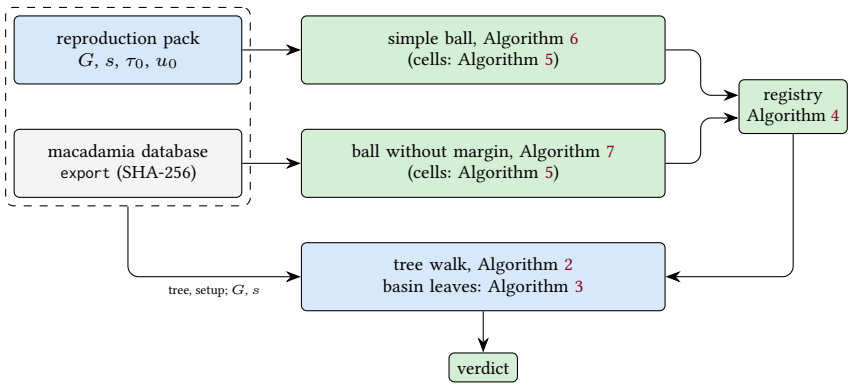


Fig. 13. The check of a form (Algorithm 1). The form and its stored chain come from the reproduction pack, the certificate from the macadamia database. Blue: code of the reproduction pack (the walk with its one added branch); green: code of this paper.

verifier changes (the other parts of the patch concern the search, the list of fields that the pack writes for a leaf, to which they add `"gc"`, `"gk"`, `"gr2"`, `"bc"`, `"bk"`, `"br2"`, and a diagnostic printed when a tree fails), and the file of the pack is never edited; a reader can compare the patched function with the original line by line.

The values of the leaves of the pack are computed by the functions of Table 2, exactly as in [6, Sections 5.7–5.9]: rational certificates, integer arithmetic, and the tests $\mathrm{ge}_{d_{\mathrm{cut}}}$ and ge_{972} of the pack. None of them calls the function `empty_region_test` discussed in Appendix A.

4.4 The basin-leaf test

Algorithm 3 (Appendix C) tests a basin leaf: it accepts the leaf if the 512 vertices of its box lie in one registered ball.

Lemma 28 (basin leaves). *If Algorithm 3 accepts a basin leaf, every chain over P whose first gluing has a representative in the box has $d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$.*

Proof. The function $t \mapsto P[\tau(t) - c_i - k]$ is a convex quadratic function of t, so its maximum over the box is reached at a vertex: every τ of the box satisfies $P[\tau - (c_i + k)] \leq r_i^2$. The centre $c_i + k$ is an integer translate of a registered centre, so $(c_i + k, r_i)$ satisfies (B) (Section 3.2). □

The index `"gc"` is a hint: the test first tries the ball it names, and, if that fails, the eight registered centres nearest to the centre of the box (in floating point), each with its own certified radius (a saved index need not point to the ball that contains the box). Either way the decision is the exact test of line 5 of the algorithm, and soundness does not depend on the order of the centres. A leaf written with the key `"basin"` is checked in the same way, against the registry of general balls only, with its index ignored: it is accepted if and only if its box lies in a general ball. The leaf `"cwit"` of the fixed-N search, which holds only for $Q = P$, is rejected.

4.5 The registry of balls

Algorithm 4 (Appendix C) builds the registry of balls of a form: the ball at the stored gluing and its images. Only balls accepted by Algorithms 6 and 7, that is balls of the general theorems, enter the registry. The walk finds the registry of a form through the set C of minimal vectors of its own reduced form. Since P is perfect, C and $\min P = 4$ determine P; so the registry and the walk work with the same matrix P, even when the setup and the head of the tree were written separately.

4.6 The Delaunay subdivision

Both kinds of ball rest on the exact Delaunay subdivision of the stored lattice M_0, of rank 10. The certificate lists the cells modulo the translations of the lattice and the map $x \mapsto -x$, each by its centre (canonically in $[0,1)^{10}$), its squared radius and its vertices, in the basis given by an integer matrix U_{red} with $|\det U_{\mathrm{red}}| = 1$, in which the Gram matrix of the lattice, G_0, multiplied by an integer D_A, is an integer matrix A. Algorithm 5 (Appendix C) checks this list.

Lemma 29 (closure). *If Algorithm 5 accepts and the facet lists of its line 6 are complete, the listed cells with their translates and opposites are all the Delaunay cells of the lattice, and ρ^2 is its squared covering radius.*

Proof. Lines 4–5 show that every listed cell is a Delaunay cell. For a facet f of a listed cell, line 9 shows that the cell across f computed by the flip is a listed cell with the same vertex set, so it is a Delaunay cell that contains f and lies on the other side of f: the neighbour across f (the tiling is face to face, and the neighbour across a facet is unique). The floating-point guess of the flip only helps to find it. So the set of listed cells, closed under translations and $x \mapsto -x$, is closed under crossing facets; the cells of a face-to-face tiling of $\mathbb{R}^{10}$ are connected through their facets, so it is the whole subdivision. Since ρ^2 is the largest squared radius of the listed cells, it is the squared covering radius by Lemma 5. □

The completeness of the facet lists is not tested by Algorithm 5: it is proved by the separate exact check of Section 4.1 (Computational result 31). The same lists give the inradius $R_{0,D}$ of a deep cell (Section 3.3), which is the least distance from p_D to the hyperplanes of the facets: a missing facet could make it too large. So the deep cells of `0014` and `0059` rest on that check as well.

4.7 The check of a simple ball

Algorithm 6 (Appendix C) checks a simple ball: it tests the hypotheses of Theorem 19. A vertex constant or a deep datum that fails its check is not a reason to reject: the checker falls back to the weaker bound ($b\,R_D^2$, or the shallow row), and the final test of line 17 decides with what is left. The tests of lines 2 and 14 are needed for soundness. Lemma 15 holds for $r \geq 0$ and for vectors of $\mathbb{Z}^9$: a negative r would make every φ_j and hence U too small, and a vector with a tenth coordinate would add to the denominator of $U(r)$ without adding to $S_9(W)$. The checker makes both tests (`frame_U`, `verify_simple`,

`register_general`, and the readers of `l12gsharp.py`), and the stored certificates pass them. The vertex constants are recomputed from the stored certificate data ($\tilde{\mu}_v$, the weights q_c) with the constant K of the whole cone, never read as stored numbers. The checks of the deep data are: $\pi \geq 0$, $\sum \pi = 1$, the mean and the second moment equation by equation, antipodality, the central radius (Section 3.3) and the inradius from the exact facets. The sum of the cell volumes is computed in floating point and must be 1; this test can only reject.

4.8 The check of a ball at a lattice without margin

Algorithm 7 (Appendix C) is Corollary 25: lines 5–7 give the contacts and the constants K, a and b of the whole cone of G_0, line 9 the margin at $t = 0$, line 10 the cells, lines 11–12 the frame and the localisation, and line 13 the covering bound. The constants of the cells (θ_2, ℓ, the distributions of Lemma 24) are recomputed exactly from the cells; floating point only proposes θ_2, ℓ, $\bar{\sigma}$, β and t_L, each accepted by an exact test. Unlike Algorithm 6, nothing in it depends on vertex constants or on linear programs. The verification record states whether the cells were checked; a record made with the cells skipped is refused by the tree stage and by the database.

4.9 The final verification on several processes

The tree of `0015` has 1463716 leaves, and its exact verification is run on several processes. The program `l12pverify.py` gives the verdict of Algorithm 2 on N processes (Figure 14); with $N = 1$ the serial walk runs unchanged. It decides no leaf itself:

- the parent checks the head as in Algorithm 2, then walks the tree in the same depth-first order *without* the leaf checks: every inner node must cut its box ($0 \leq v < 9$, $\ell_v < x < h_v$), an unfinished node rejects, and the leaves are numbered in the order of the walk;
- every leaf, with its box, is sent to a worker. The workers are forked after every patch of the walk and every registration of a ball, so they hold the same verifier state as the parent; each first sends a fingerprint of that state, which must equal the parent's. A worker checks a leaf by calling the patched leaf branch of the walk on that leaf alone;
- the tree is accepted only if every leaf index comes back exactly once, as accepted and with the kind that the parent read, every worker ends normally, and the number of leaves equals the number recorded in the head, when the head records one (the heads of the five trees do, and the database compares the census of their leaves with the stored one). Any exception, death of a worker, timeout, missing, repeated or unknown index rejects. Results travel through pipes only.

Since every patch of the walk changes the branches of the leaves only, the parent's walk of the inner nodes is the serial one, and the verdict is the serial verdict. A leaf read lazily from the file is turned into the same object as one read in memory, so the two ways of reading give the same verdict. The database program passes the number of processes to the tree stage and records it (Section 5.3).

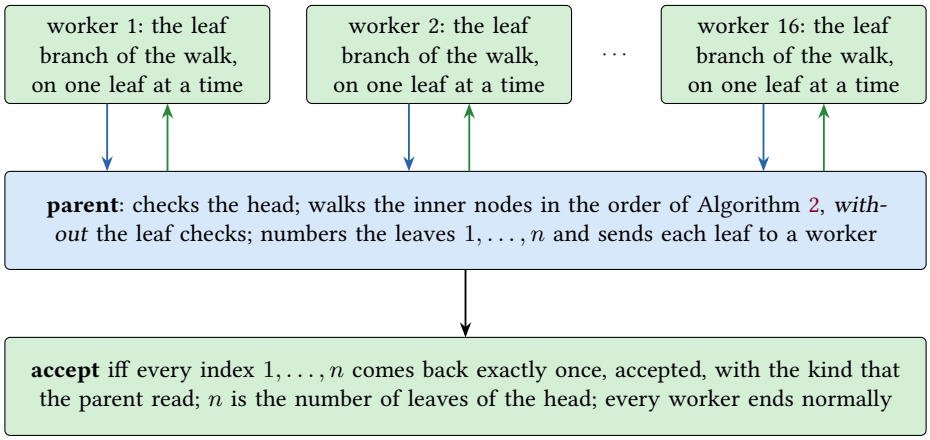


Fig. 14. The parallel final verification. The parent decides the shape of the tree and the workers decide the leaves, with the unchanged leaf branch. Green arrows: the parent sends a worker a leaf i and its box. Blue arrows: a worker first sends the fingerprint of its verifier state, which must equal the parent's, then, for each leaf, (i, kind, accepted). The workers are forked after every patch of the walk and every registration of a ball, so they hold the same verifier state as the parent. The tree is accepted only if every leaf comes back exactly once; any exception, death of a worker, missing, repeated or unknown index rejects.

5 The database

5.1 Format and scopes

The certificates are stored in one SQLite file, the database of record `macadamia_certificates_cg5.sqlite`. Its format is format 2 of the series with the scope extension and the extensions of the program `macadamia_db_v6_3.py`, which writes and reads it. Every certificate, every ball and every stored file has a *scope*:

- `general`: it holds for every N admissibly marked by P. A general certificate is a tree checked in general mode, with its balls;
- `fixedN`: it holds for $N = P$ only. The general check never reads a row of this scope.

Only certificates count. The program can hold a row of the scope `general` that only cites a proof made elsewhere (an *external* row); it lists such a row apart, never counts it, and its `verify` of such a row exits with the status 2 and the words "not counted". When a certificate replaces an external row, the certificate goes into a table of its own and a marker row records the replacement; the external row stays unchanged, as history. No result of this paper uses an external row.

Table 5 lists the tables, Figure 15 shows how they are linked, and Table 6 the roles of the stored files of a general certificate. Rationals are stored as JSON pairs $[p, q]$.

The general balls of the five certificates have three *theorems*, read from the ball file itself and never guessed: `G-quadratic` (kind `gsimple`, Theorem 19 with the constant K of Lemma 9 `b`; checked by `l12general.verify_simple`), `G-cubic` (kind `gsimple`, with the constant K of Lemma 10; checked by `l12general2.verify_simple`) and `S` (kind `gsharp`,

Table 5. The tables of the database. The keys are `file:row` names of forms; `blob` holds the bytes of a file.

table	columns and use
`meta`	the format and one row per extension (number, date, tool)
`cert`	one row per certificate, key `(key, scope)`: `name`, `stored_utc`, `method`, `proof`, `tree_kinds` (leaves by kind), `tree_sha256`, `results` (the record of the tree run), `verdict`, `code_zip_sha256` and `code_zip_name` (the build to verify with), `switches` (the switches of the run), `extras` (the census of the leaves and the modules needed), and columns for citations
`cert_superseding`, `superseded`	a certificate that replaces an external row, and the marker of the replacement
`form`	the rows of format 2 of the fixed-N certificates, unchanged
`ball`	one row per ball: `kind`, `theorem`, `tau_c`, `u_c`, `r`, `scope`, `cert_file` and `cert_sha256` (the ball file and its SHA-256), `record_file` and `record` (its verification record), `data_sha256` (the SHA-256 of the data files that the ball file names)
`file`	one row per stored file: `role`, `filename`, `relpath` (its path in the output folder of the run), `bytes`, `sha256`, `codec` (`zlib` or none), `blob`; size and SHA-256 are those of the file itself, before compression
`verify_log`	one row per recorded verification: form, scope, date, host, tool and its SHA-256, the build used and its SHA-256, the number of processes, the result

Table 6. The roles of the stored files of a general certificate, with their paths in the output folder of the run (`NAME`: the key of the form with : replaced by _; `CASE`: its short name).

stored role	files (`relpath`)
`'general tree'`	`trees/NAME.tree.json.gz`: the tree of boxes; its SHA-256 is `tree_sha256`
`'general tree setup'`	`trees/NAME.setup.json`: its setup
`'general tree result'`	`trees/NAME.result.json`: the state of the search (`excluded`)
`'general tree results'`	`tree_results.json`: the verdict of the run, kept as it was
`'general simple ball'`	`CASE/CASE.gsimple.json` and its record `CASE/CASE.gsimple.verify.json`, with the cone of P (`CASE.pcone.json.gz`), the reference data (`CASE.basin.json.gz`, or `CASE.basin2.json.gz` for a K of degree three) and its cells, vertex constants and deep data
`'general sharp ball'`	`CASE/CASE.gsharp.json` and its record `CASE/CASE.gsharp.verify.json`, with its cell file `CASE.gsharp.cells.json.gz` and its cone file `CASE.gsharp.cone.json.gz`

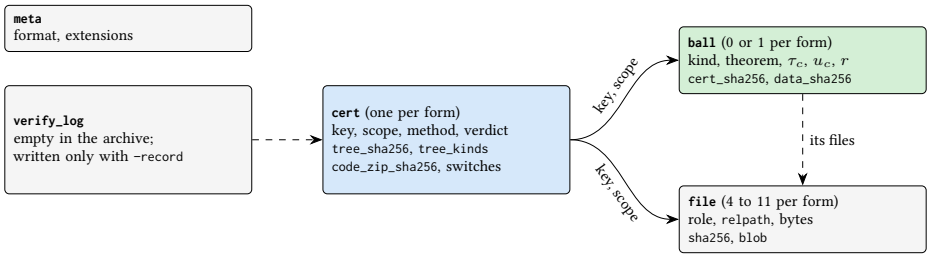


Fig. 15. The tables of the database of record. A certificate is one row of `cert`, its ball one row of `ball`, and every file is stored with its SHA-256; `verify` exports the files, checks every SHA-256 and verifies the form again.

Corollary 25; checked by `l12gsharp.verify_ball`, Algorithm 7). The program knows further kinds of ball and of leaf, which none of the five certificates contains.

5.2 Storing a general certificate

A general certificate enters the database from the output folder of the run that produced and verified it, by the command `add-general CASE OUT --code-zip ZIP`. Algorithm 8 (Appendix C) is its *acceptance rule*: if one of its tests fails, nothing is written. Every test and every stored SHA-256 [14] is computed on the same bytes. The rule checks that the run recorded a valid result for exactly the files that are stored; the mathematics is verified again by `verify`.

The append-only rule. No command of the program changes or deletes a row. A form with a general certificate cannot receive another one; every command writes in a single transaction, so a refused or interrupted command leaves the database as it was. The command `upgrade` adds the extensions to an older database: it first makes a backup with the backup interface of SQLite, compares it with the database table by table, then works in one transaction and checks by digests that every earlier row and value is unchanged. The database of record is the file whose SHA-256 is published with the archive (Section 6).

5.3 Export and verification

Algorithm 9 (Appendix C) exports a form: it rebuilds the output folder of the run. The command `verify CASE --scope general --pack PACK --code-zip ZIP [--verify-processes N] [--record]` then does the following.

1. It exports the form into a new folder (Algorithm 9).
2. It applies the acceptance rule of Algorithm 8 to the export and compares the result with the rows: files, balls (kind, theorem, radius, SHA-256), tree SHA-256, leaf kinds, switches. It moves `tree_results.json` aside, so that the verdict can only come from the new run.
3. It checks the build: the SHA-256 of `ZIP` must be the recorded one, and every module needed must be in it.

Table 7. The general certificates. The leaves of the kinds `box`, `pattern`, `pspec` and `uspec` prove $d \geq d_{\mathrm{cut}}$, those of kind `three` prove $\Delta \geq 972$, those of kind `gball` lie in the ball of the form.

form	leaves	outside	box	pattern	pspec	uspec	three	gball
0001	13696	4012	456	5239	1783	1694	0	512
0014	9670	3165	174	78	1134	3924	0	1195
0059	145815	69090	1405	265	10246	45332	0	19477
0015	1463716	464021	1118	0	13453	692863	523	291738
0052	87470	36938	3255	0	5787	40846	644	0

4 It verifies the ball again, exactly, from its data and from the row of the pack, with the checker of its theorem (Algorithm 6 or 7). The new record must be valid, with the stored radius and SHA-256.

5 It runs the tree stage of the checker in general mode with exactly the stored switches, in an environment cleaned of every other switch, and with N processes for the final verification (Section 4.9).

6 It requires that the log shows general mode, the registration of as many balls as are stored, and a valid exact verification of the tree, and that the new `tree_results.json` has `"tree_valid"` and `"general_mode"` true, the stored leaf kinds, the stored SHA-256 of the tree and the number N.

The exit status is 0 for a valid certificate, 1 for a failure, and 2 when nothing could be checked: the pack, the build or the Python is absent, or the row is an external one, which is not counted. With `--record` the result goes into `verify_log`, with the SHA-256 of the build actually used.

5.4 What is stored

The database of record, `macadamia_certificates_cg5.sqlite` (958324736 bytes; SHA-256 in Section 6), holds exactly the general certificates of the five forms, each with its tree, its setup, its records and, except for `0052`, the files of its ball, and nothing else. Each certificate was stored by `add-general`, so that the whole acceptance rule of Algorithm 8 ran on it. Table 7 gives the numbers of the five certificates, Figure 16 the shares of the kinds of leaves, and Figure 17 their sizes and the times of their checks.

The trees of `0001`, `0014`, `0059`, `0015` and `0052` have 27391, 19339, 291629, 2927431 and 174939 nodes and depths 37, 40, 88, 61 and 42; the exact verification of the first three took 185 s, 233 s and 3788 s on one process; Section 6 gives the times of the whole check of each form, `0052` included. The SHA-256 of the tree files are

```
0001 172854a75db88088e2d9df6c7a2990c656528a9b010c6eb00fb9c7e6cbf484f7
0014 601b46b9cd85870be5893eefe7cb868ee79422db29de6a9637bd6d544fd45499
0059 1c3425187ada58fae43a3b94351629cc7506cdfd74b3140f40ab91f722dfe93a
0015 d6e2f725a7a24f4d94ca1b4e120e40a4cd319ff16e887e4add6ba956bbc49e9b
0052 406717f6d0bbaa6de754a013418d44137c181d534b5597a1be63f8ab58dfce21
```

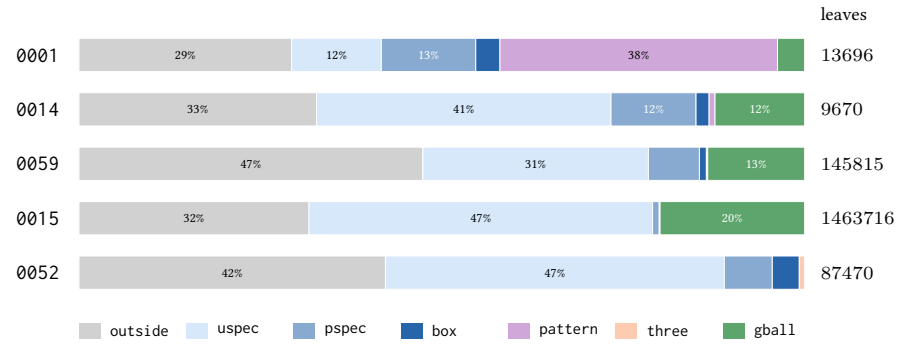


Fig. 16. The leaves of the five trees by kind, as shares of each tree (Table 7). The basin leaves (`gball`) are the part of a tree that rests on the ball of the form: a fifth of the tree of `0015`, none for `0052`.

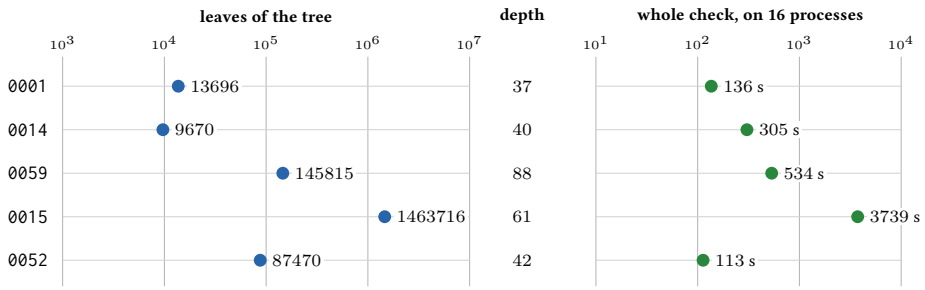


Fig. 17. The size of the five certificates, on logarithmic scales: the leaves of each tree, its depth, and the time of the whole check of the form (export, ball and tree) on 16 processes. The tree of `0015` has more than a hundred times the leaves of the smallest trees.

6 How a reader checks the certificates

The archive. Everything needed is in the Zenodo record [12]. Its file `cg5_zenodo_packet_v5_20261117.tgz` holds eleven files:

- the database of record `macadamia_certificates_cg5.sqlite` (Section 5.4);
- the program `macadamia_db_v6_3.py` (commands `list`, `export`, `verify`);
- the build `l12_build_cg5.zip` of the checkers, which holds `l12general.py` (the simple balls, Algorithm 6, the leaf `gball` and its registry), `l12general2.py` (the same with Lemma 10), `l12gsharp.py` (Algorithm 7), `l12k11.py` (the identities of degree three and four and of Lemma 11), `l12verify.py`, `l12delaunay.py`, `l12local.py`, `l12ref.py`, `l12exact.py` (exact primitives and the checks of the cells), `l12tree.py` (the run-time patch of the verifier of the pack), `l12run.py` (the tree stage) and `l12pverify.py` (Section 4.9), with their tests;
- `install_env.sh`, which prepares a machine, and `verify_all.sh`, which checks the five forms and then the facet lists;

– the folder `cellcheck`: the exact check of the completeness of the facet lists of Section 4.1 (`check_cell_completeness.py`), its test and the script `check_cells.sh` that runs it;
– `pack_reference.env`, the identity of the reproduction pack (DOI, address, SHA-256 of its zip, of its certificate database and of its `k11lib.py`);
– a `README.md` and the file `SHA256SUMS` of the ten other files.

Their SHA-256 are:

the packet	`e77b1021162f728deca8d5a909212d7585fabc291fb9529eb14335bf4bb5aff5`
`SHA256SUMS`	`bd2ac5c9b1d19ec606a38aaecd4ed2f623f768a747c86265aa7a08f86948a37d`
the database of record	`3864c8394f751f43c91e0e33bcd2acc85c99a27a1c78150752e56605defe00bf`
the build	`3952dda4dc052e6ee18cc716a34fa138007c8c8645f2052d1586da8d8ad64fd7`
the cell check	`1ce52186fdd538643b7bcef5444e04997d675994d8a3e196c01a1cf782647900`

The second file of the record, `cg5_integer_extension_v3_20261117.zip`, holds the extension of Appendix A, its test and the count of the leaves of kind `empty`; the check does not use it. The reproduction pack [6] is not in the record: `install_env.sh` downloads it from its own record, doi:10.5281/zenodo.23979316, and checks it.

The environment. A Linux machine with 16 cores, 64 GB of memory and 20 GB of free disk. The script `install_env.sh` installs the system packages (a C compiler, the GMP headers, `curl`, `unzip`), downloads the reproduction pack and checks it (the MD5 published by its record, the SHA-256 of its zip, of its certificate database and of its `k11lib.py`), builds the Python environment of the pack (in our run: Python 3.12.16, numpy 2.5.5, scipy 1.18.1, fpylll 0.6.4; the versions are not pinned) with pycddlib 2.1.7 for the exact double description (GMP rationals), and runs the self-test of the checkers. The checks use numpy, scipy and fpylll only for the guesses, the diagnostics of Section 4.1 and the guarded integer arithmetic of Lemma 10; every decision is taken with Python's integers and fractions.

The commands. After unpacking, three commands check the five certificates:

```
$ tar xzf cg5_zenodo_packet_v5_20261117.tgz && cd cg5packet
$ bash install_env.sh
$ bash verify_all.sh
ALL 5 CERTIFICATES VALIDATED
```

The script `verify_all.sh` checks every file against `SHA256SUMS`, runs the self-test of the checkers again, then runs, for `0001`, `0014`, `0052`, `0059` and `0015` in turn, the command of Section 5.3 on 16 processes, and finally the check of the facet lists:

```
$ python3 -I tool/macadamia_db_v6_3.py --db macadamia_certificates_cg5.sqlite
    verify 0014
    --scope general --pack PACK --code-zip l12_build_cg5.zip
    --python PYTHON --workdir DIR --processes 16 --verify-processes 16
RESULT: 0014:001749549 -- general certificate VALID: ...
```

Each run re-verifies the ball (Theorem 19, Lemma 10 for `0059`, Corollary 25 for `0015`; none for `0052`) and walks the stored tree again. A form counts as valid only if the command ends with status 0, prints its line `RESULT`, and its tree log shows the exact verification on 16 processes; the last line is printed only when the five are valid and

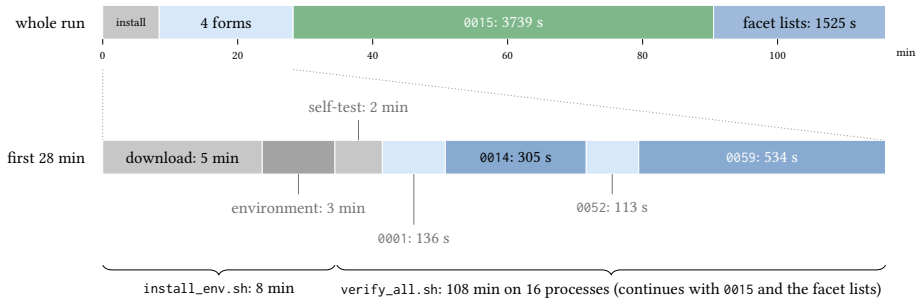


Fig. 18. The whole check on an empty system: 116 minutes, of which 8 for the installation and 108 for the five forms and the facet lists on 16 processes.

the facet lists complete, and the database is never written. The whole procedure was run from an empty Ubuntu 22.04 system, on which it printed the last line above: the installation took 8 minutes, including the download of the pack, and the verification 108 minutes on 16 processes (136 s for `0001`, 305 s for `0014`, 113 s for `0052`, 534 s for `0059` and 3739 s for `0015`, each time for the whole command: export, ball and tree; then 1525 s for the facet lists; Figure 18).

Running times. On two cores, the exact check of a simple ball took about one hour, most of it for the Delaunay cells (7070 cells with 320104 facets for `0014` and `0059`); the check of the ball of `0015` took 3995 s (7302 cells with 317703 facets). The exact verification of the trees took the times of Section 5.4.

7 What is proved

Computational result 30 (the check). For each of the forms `0001`, `0014`, `0059`, `0015` and `0052`, the checks of Algorithm 1 succeed on the certificate of the form and on the row of the form in the certificate database of the reproduction pack:

a the ball, if any, is accepted, with the Delaunay cells checked again: by Algorithm 6 for `0001` ($r = 64497/200000$), `0014` ($r = 249731/10^6$) and `0059` ($r = 343459/10^6$, with the constant $K = 5/2$ of Lemma 10), and by Algorithm 7 for `0015` ($r = 251891/10^6$, $K = 47/14$, $a = \frac{1}{2}$, $T^* = 0.202775$); the certificate of `0052` has no ball;

b the exact verification of the tree in general mode (Algorithm 2, with the registry of Algorithm 4 built from the ball and its images, or empty for `0052`) is valid (Table 7).

For the five forms both checks were run again by `verify --scope general` from the database of record, the build and the reproduction pack, by the procedure of Section 6 on an empty system.

Computational result 31 (the facet lists). For the stored lattices of `0001`, `0014`, `0059` and `0015`, the exact check of Section 4.1 succeeds: the facet lists of the 25798 Delaunay cells, 1516271 facets in all, are complete, and they are the lists that the checkers compute. The script `verify_all.sh` runs this check with the others.

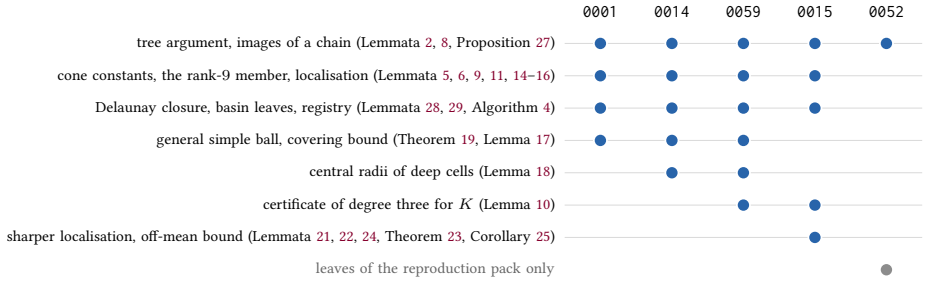


Fig. 19. The theorems of this paper that each certificate uses (part (b) of the trust base). The certificate of `0052` uses the tree argument only: its leaves are those of the reproduction pack.

Trust base. The five certificates, and Theorem 32 below, rely on the following.

(a) Results of the root paper and of the focus paper, assumed.

- **A1** The reduction of γ_{11} to chains $L \supset M \supset N$ over the perfect forms of rank 9 that survive the selection of [5]: [5, Proposition 15] and the admissible marking of [5, Proposition 7 and Section 6.2], as used in Definition 1.
- **A2** The conditions on a chain used by the leaves of the pack and by the ball theorems: [6, Section 2.4], that is **(M)**, **(L2)**, **(Mk)** (a consequence of the condition (Domain) of [6]) and **(u)** of Section 2. The last one rests on the value of γ_{10} [5, Theorem 3], with its hypotheses, applied to the dual of M.
- **A3** Lemma 3 ([6, Lemma 41]), Lemma 4 ([6, Lemma 42]), the inequality of Lemma 12 ([6, Lemma 37]) and Lemma 13 ([6, Lemma 36]); their proofs are in [6].
- **A4** What each kind of leaf of the pack proves on its box ([6, Table 1 and Sections 5.5–5.9]), the tree argument ([6, Lemma 11 and Proposition 18]), and the root box, the symmetry data and the leaves of kind `outside` ([6, Section 5.7 and Proposition 15]).

(b) Theorems of this paper that the checkers implement (Figure 19).

- **B1** All five certificates: Lemmata 2 and 8, and Proposition 27.
- **B2** `0001`, `0014`, `0059` and `0015`: Lemmata 5, 6, 9, 11, 14, 15, 16, 29 and 28, the monotonicity of the minorant (the last sentence of Lemma 12), and Algorithm 4.
- **B3** `0001`, `0014` and `0059`: Theorem 19 with Lemma 17; `0014` and `0059` also Lemma 18 (its item `b` is the option stored); the ball of `0001` uses no deep cell and no vertex certificate.
- **B4** `0059` and `0015`: Lemma 10 ($K = 5/2$ and $K = 47/14$, of degree three).
- **B5** `0015`: Corollary 25, with Lemmata 21 and 22, Theorem 23 and Lemma 24.
- **B6** `0052` uses no theorem of this paper beyond **B1**: its tree has leaves of the pack only.

(c) Software, data and hardware.

- **C1** The exact verifier of the reproduction pack, `verify_box_certificate` of `k11lib.py` and the functions of Table 2, with the branch for the basin leaves

added at run time (Section 4.3); the published `k11lib.py` is used unchanged: the function extended in Appendix A is one that the walk does not call.

- **C2** The exact verifiers of this paper in the build of the archive (Section 6), including the parallel final verifier of Section 4.9 when more than one process is used.
- **C3** The database program `macadamia_db_v6_3.py` and the script `verify_all.sh`, which run Algorithms 1, 8 and 9 and give the verdict (Sections 5.3 and 6). The check of a ball and the walk of the tree are two processes: the tree stage registers a ball from the record that the check of the ball wrote, which names the ball file by its SHA-256, and the verdict is read from the result file and from the log of the run.
- **C4** The Python interpreter: its integers and `fractions.Fraction`, and the parts of its standard library that read the files, start the processes and install the branch of the basin leaves. NumPy 64-bit integers only where an explicit bound excludes overflow (in the pack, and in the check of Lemma 10); floating point never decides anything.
- **C5** cddlib, to this extent only: it proposes the facets of a cell. Each facet is rechecked exactly, and the completeness of the lists is proved by the separate program `check_cell_completeness.py` (Section 4.1), which is trusted as the checkers are. That program certifies the lists of its own run; it is assumed that cddlib returns the same list for the same cell in the run of the checkers.
- **C6** SQLite keeps the files, and SHA-256 identifies them: it is assumed that no two different files met by the checks have the same SHA-256.
- **C7** The data: the perfect forms of rank 9 of [3]; the forms and the stored chains (τ_0, u_0) read from the certificate database of the reproduction pack [6, Section 6], which also defines the list of the macadamias.
- **C8** The hardware and the operating system on which the checks run.

Not relied on. The case-by-case proofs of [7] and the computer proof of [6, Appendix E]. The certificates check these proofs again (Table 8); they do not use them. The two routes share the toolbox of part (a): an error in one of the four lemmata quoted from [6], or in the reduction of [5], would affect both.

Theorem 32. *Assume Computational results 30 and 31. Let P be one of the forms* `0001`, `0014`, `0059`, `0015` *and* `0052`. *Every chain $L \supset M \supset N$ of [5, Proposition 15] whose rank-9 member N has a Gram matrix admissibly marked by P has $d \geq d_{\mathrm{cut}}$ or $\Delta \geq 972$.*

Proof. Such a chain is a chain over P (Definition 1). By Computational result 30 the exact verification of the certificate of P is valid, and by Computational result 31 the facet lists are complete. So Proposition 27 applies: (E_P) holds, which is the claim. □

Theorem 32 is the statement that [5, Section 6.2] requires for each of these five macadamias. Table 8 lists the proofs that the five certificates check again.

8 Use of computational tools

The use of computational tools, artificial intelligence included, and the computing facilities are those described in the first paper of this series [5].

Table 8. The proofs that the five certificates check again: the whole of [7] and Appendix E of [6] (pages 51–56).

form	proof	pages
0001, 0014, 0059, 0015	the whole paper [7]	36
0052	[6, Appendix E] (computer proof)	6
total		42

References

1. Delaunay, B.: Sur la sphère vide. Izv. Akad. Nauk SSSR, Otd. Mat. Estestv. Nauk **7**, 793–800 (1934)
2. Conway, J.H., Sloane, N.J.A.: Sphere Packings, Lattices and Groups, 3rd edn. Springer, New York (1999). https://doi.org/10.1007/978-1-4757-6568-7
3. Dutour Sikirić, M., van Woerden, W.: The lattice packing problem in dimension 9 by Voronoi's algorithm. arXiv:2508.20719v2 (2026). Data: https://doi.org/10.5281/zenodo.15707640
4. Fukuda, K., Prodon, A.: Double description method revisited. In: Combinatorics and Computer Science, LNCS, vol. 1120, pp. 91–111. Springer, Berlin (1996). Software: cddlib, https://github.com/cddlib/cddlib, and its Python interface pycddlib
5. Géraud-Stewart, R., Naccache, D.: The Hermite constants in dimensions 10, 11 and 12. Cryptology ePrint Archive, Paper 2026/2825 (2026), https://eprint.iacr.org/2026/2825
6. Géraud-Stewart, R., Naccache, D.: A focus on Hermite constant 11. Cryptology ePrint Archive, Paper 2026/2826 (2026), https://eprint.iacr.org/2026/2826
7. Géraud-Stewart, R., Naccache, D.: γ_{11}: Contact geometry — the macadamias 0001, 0014, 0059 and 0015. Cryptology ePrint Archive, Paper 2026/2827 (2026), https://eprint.iacr.org/2026/2827
8. Géraud-Stewart, R., Naccache, D.: γ_{11}: A finite spectral relaxation — the macadamia 0008. Cryptology ePrint Archive, Paper 2026/2828 (2026), https://eprint.iacr.org/2026/2828
9. Géraud-Stewart, R., Naccache, D.: γ_{11}: Reduction by sections — the macadamias 0019 and 0017. Cryptology ePrint Archive, Paper 2026/2829 (2026), https://eprint.iacr.org/2026/2829
10. Géraud-Stewart, R., Naccache, D.: γ_{11}: One selected contact, two linked blocks — the macadamia 0030. Cryptology ePrint Archive, Paper 2026/2830 (2026), https://eprint.iacr.org/2026/2830
11. Géraud-Stewart, R., Naccache, D.: γ_{11}: Unequal triangles and adapted references — the macadamia 0050. Cryptology ePrint Archive, Paper 2026/2849 (2026), https://eprint.iacr.org/2026/2849
12. Géraud-Stewart, R., Naccache, D.: γ_{11}: computational certificates. Referee packet: database of record, checkers and scripts. Zenodo (2026). https://doi.org/10.5281/zenodo.24269559
13. Martinet, J.: Perfect Lattices in Euclidean Spaces. Grundlehren der mathematischen Wissenschaften, vol. 327. Springer, Berlin (2003). https://doi.org/10.1007/978-3-662-05167-2
14. National Institute of Standards and Technology: Secure Hash Standard (SHS). FIPS PUB 180-4 (2015). https://doi.org/10.6028/NIST.FIPS.180-4

A The extension of an integer test

The authors attract the reader's attention to a detail in the function `empty_region_test` of `k11lib.py`, the library of the reproduction pack. This appendix states the detail in question, why no published result is affected, and the function's extension, which the archive of this paper supplies.

The function. A region of a tree of cases is a vector of bounds $a_i/64 \le \|\theta(z_i)\| \le b_i/64$ on a pool of vectors $z_i \in \mathbb{Z}^9$, where θ is the first gluing. A leaf of kind `empty` names nine pool vectors that form a basis of $\mathbb{Z}^9$, so that $z_i = \sum_j c_{ij} z_{S_j}$ with integers c_{ij}; for each of the 256 choices of signs e_j (with $e_1 = +1$), $\theta(z_i) = \sum_j c_{ij} e_j t_j$ with $t_j \in [a_{S_j}, b_{S_j}]/64$ lies in an interval $[L_i, H_i]/64$, and the test discards the sign choice if for some i that interval contains no x with $a_i/64 \le \|x\| \le b_i/64$. The region is called empty if all 256 sign choices are discarded [6, Section 5.6].

The observation. The shipped function converted the integers c_{ij} to NumPy `int64` arrays and computed all the endpoints L_i, H_i by two matrix products. These products wrap modulo 2^{64} without an exception once $32 \sum_j |c_{ij}|$ exceeds 2^{63}, and the wrapped endpoints can come out in the wrong order. Here is a theoretical counterexample having the identity basis and the 256 vectors $(K, \pm K, \ldots, \pm K)$ with $K = 2^{58}$, 265 pool vectors in all, the bounds $[31, 32]$ on the basis and $[0, 1]$ on the other vectors. The rational gluing $\theta = (\frac{63}{128}, \ldots, \frac{63}{128})$ satisfies every bound: $\|63/128\| = 31.5/64$, and every other $z_i \cdot \theta$ is an integer. For its sign choice and the vector $(K, \ldots, K)$, the exact interval is $[279K, 288K]$; the `int64` products give $23K$ and -2^{63}. The sign choice is discarded, and the shipped function calls the region empty. With $K = 2^{20}$ and $K = 2^{50}$ there is no overflow and the answer is right.

No published result is affected. The function is called by the exact verifier only for leaves of kind `empty`, and only in trees of cases (`verify_tree_certificate`); the walk of a tree of boxes (Algorithm 2) never calls it, so no certificate of this paper does. In the published certificate database of the reproduction pack (`k11_certificates_v3.sqlite`, 4 487 813 rows, SHA-256 `982eaf73. . . c59e8`), no tree has a leaf of kind `empty`: the token `empty` occurs on no line of any of its 70 trees. The program `count_empty_nodes.py` of the archive counts them on the published file, which it only reads; it prints `70 trees; 0 lines with the token empty`.

```
import sqlite3, gzip, io, sys
if len(sys.argv) != 2: sys.exit('usage: python3 -I count_empty_nodes.py PATH/TO/k11_certificates_v3.
      sqlite')
con = sqlite3.connect('file:%s?mode=ro' % sys.argv[1], uri=True); trees = lines = 0
for (cert,) in con.execute("SELECT cert FROM form WHERE kind = 'tree'"):
    trees += 1
    lines += sum(b'empty' in line for line in gzip.open(io.BytesIO(cert)))
print(trees, 'trees;', lines, 'lines with the token empty')
```

The observation therefore never applied. For the other uses of `int64` arithmetic we searched the 53 functions of `k11lib.py` that `verify_box_certificate` can reach, found by following every name of a function that a function mentions. Arrays of type `int64` occur in three of them: `_box_vertices`, a matrix of zeros and ones, and `_box_leaf_core`

and _box_rows_exact (used by three_leaf_value), which take the int64 path only after an explicit bound shows that no intermediate value can overflow, and use Python integers otherwise. The symmetry data and the leaves outside are checked with Python integers and fractions only.

The extension. The extended function computes the same endpoints with Python integers, which are unbounded:

```
for p in range(2 ** (N - 1)):                    # the 256 sign choices, e_1 = +1
    sg = [1] + [1 if (p >> j) & 1 == 0 else -1 for j in range(N - 1)]
    lo = [aa[S[j]] if sg[j] > 0 else -bb[S[j]] for j in range(N)]
    hi = [bb[S[j]] if sg[j] > 0 else -aa[S[j]] for j in range(N)]
    alive = True
    for i in range(k):
        ci = Cc[i]
        Llo = sum(c * (lo[j] if c > 0 else hi[j]) for j, c in enumerate(ci))
        Lhi = sum(c * (hi[j] if c > 0 else lo[j]) for j, c in enumerate(ci))
        plus = -((-(Llo - bb[i])) // U) <= (Lhi - aa[i]) // U
        minus = -((-(Llo + aa[i])) // U) <= (Lhi + bb[i]) // U
        if not (plus or minus): alive = False; break
    if alive: return False                       # this sign choice may hold a gluing
return True
```

The file cg5_integer_extension_v3_20261117.zip of the archive contains it in three forms: the patch k11lib_empty_region_exact.patch, which changes only this function in k11lib.py (it applies to the file of the pack, SHA-256 6bb677dc. . . ad8b4); the module k11lib_empty_region_exact.py, whose function install replaces the function of an imported k11lib.py; and the file k11lib.py of the pack, already patched. The check of the five certificates does not use it. The test test_empty_region_overflow.py reproduces the counterexample and checks the extension:

```
$ python3 -I test_empty_region_overflow.py PACK/programs/k11lib.py
K = 2^58: theta = (63/128, ..., 63/128) lies in the region   as expected
K = 2^58: shipped (NumPy int64)   -> empty (WRONG)           as expected
K = 2^58: exact module            -> not empty               as expected
K = 2^58: patched k11lib.py       -> not empty               as expected
... (K = 2^20 and 2^50, an empty region, 300 random regions) as expected
RESULT: PASS -- the shipped test calls a non-empty region empty;
        the exact test and the patched k11lib.py are right.
```

B The layout of the stored files

The tree. NAME.tree.json.gz is ordinary JSON, laid out with one node per line so that it can be read without holding all the leaves in memory [6, Section 5.7]: a first line with the head, ending with "tree":, then the inner nodes as {"v":j,"at":x,"le": ... ,"ge": ... } and one leaf per line. A basin leaf is one line:

```
{"H":0,"kind":"gball","gc":2,"gk":[0,-1,0,0,1,0,0,0,0],"gr2":[p,q]}
```

("H" is the marker of a leaf in the verifier of the pack; for a basin leaf it carries no data); the equivalent spelling has the same line with the keys "basin", "bc", "bk", "br2". NAME.setup.json holds "basis_U", "vectors", "box_basis", "box_L", "box_DL" and, when present, "sym" with the matrices V of Algorithm 4.

The simple ball. `CASE.basin.json.gz` holds `"version"` (`l12-basin-1`), `"theorem"` (`fixedN-1`), `"case"`, `"P"`, `"tau0"`, `"u0"` (rationals as pairs), `"D_A"`, `"U_red"`, `"contacts"`, `"eutaxy"`, `"K"` (`"K_lp"`, `"alpha"`, `"rho_R"`, `"K_cert"`), `"a"`, `"a_used"`, `"relative"` (`"K_rel"`, `"a_rel"`), `"mu_top"`, `"deep_gap"`, the names of the three companion files and `"basin_r"`. The cell file holds one record per cell modulo translations and $x \mapsto -x$ (centre, squared radius in the units of A, vertices, number of facets):

```
{"n": 10, "A": [[...]],
 "cells": [{"c": [[p,q], ...], "R2": [p,q], "V": [[...], ...], "nf": n}, ...]}
```

The file of vertex constants maps each cell (its centre, as JSON) to `"mu"` and to one certificate per vertex (`"mu_lp"`, `"q"`); the file of deep data maps each deep cell to `"pi"`, `"kappa"`, `"c"`, `"m4"`, `"sos_Q"`, `"R0"`. The files `CASE.K.json.gz` and `CASE.a.json.gz` repeat the constants, and `CASE.basin_summary.json` and `CASE.verify.json` are readable summaries; the check uses neither.

The general simple ball. The file of the simple ball, `CASE.gsimple.json`, of version `l12general-simple-1`, holds `"case"`, `"P"`, `"tau0"`, `"u0"`, the names `"basin_file"` (the file `CASE.basin.json.gz` above, whose cells, vertex certificates, deep data, `"K"` and `"a"` it uses; its relative constants and its `"basin_r"` are not used) and `"pcone_file"`, the frame `"frame"` (`"z"`: integer vectors, `"q"`: rationals), the radius `"r"`, `"mu"` (`certified`, or `crude` when every vertex constant is $b\,R_D^2$), and `"proposals"` (s_1, t_L, t_1, re-checked). `CASE.pcone.json.gz` holds the data of the contact cone of P: `"contacts"`, `"eutaxy"`, `"K"` (as above) and `"a"` (a certificate for $\lambda_{\min}$, or `null`). When K comes from an identity of degree three (Lemma 10), `"basin_file"` names a file `CASE.basin2.json.gz`: the reference data above with the field `"K"` replaced by

```
{"kind": "cubic", "K_cert": [p, q], "gens": [<integer matrices>],
 "reps": [[[c1, c2, c3], p, q], ...]}
```

that is the constant, the generators of a group of automorphisms and one triple of contacts per orbit with its coefficient $\beta = p/q$. Such a file is read by `l12general2.py`.

The ball at a lattice without margin. The file `CASE.gsharp.json`, of version `l12gsharp-2`, holds `"case"`, `"key"`, `"P"`, `"tau0"`, `"u0"`, `"U_red"`, the radius `"r"`, the frame `"frame"` (`"z"`: vectors of 9 integers, `"q"`: rationals), the proposals `"beta"`, `"smax"` and `"tL"` of Lemma 21 (re-checked), the number `"ngrid"` of pieces of the cell test, the names `"cells_file"` and `"cone_file"`, their SHA-256 in `"sha256"`, and a readable `"summary"`, which the check does not use. The cone file holds `"contacts"`, the generators `"gens"`, `"K"` with its coefficients `"cubic_reps"` (one triple of contacts per orbit), and `"a"` with `"lam_reps"`. The cell file holds `"U"`, the reduced Gram matrix `"Ared"`, the contacts `"C"` and the cells (`"c"`, `"R2"`, `"V"`).

The record of a tree run. The entry of a form in `tree_results.json`, with symbolic values:

```
{"CASE": {"status": "excluded", "general_mode": true, "tree_valid": true,
          "leaf_kinds": {"outside": n1, "uspec": n2, "pspec": n3, "box": n4,
                         "pattern": n5, "gball": n6},
          "verify_seconds": t, "tree_sha256": "<64 hexadecimal digits>"}}
```

The verification record of a ball, `CASE.gsimple.verify.json` or `CASE.gsharp.verify.json`, holds the list of the checks with their results, and `"valid"`, `"r"`, `"sha256"` (of the ball file), `"key"`, `"skip_cells"` (it must be `false`) and `"version_checker"`.

The folder of a general certificate. The export of the scope `general` writes the files at the paths of Table 6:

```
tree_results.json
trees/NAME.tree.json.gz    trees/NAME.setup.json           trees/NAME.result.json
CASE/CASE.gsimple.json     CASE/CASE.gsimple.verify.json   CASE/CASE.pcone.json.gz
CASE/CASE.basin.json.gz    CASE/CASE.cells.json.gz
CASE/CASE.mu.json.gz       CASE/CASE.deep.json.gz
```

These are the 11 files of the general certificates of `0001` and `0014`; that of `0059` has `CASE.basin2.json.gz` as reference data, that of `0015` the four files of its ball (`CASE/CASE.gsharp.json`, its record, its cell file and its cone file) in place of the files of the simple ball, and that of `0052` the tree files only.

C Algorithms

The algorithms of the checkers, in the order of the text; each is referred to where it is used.

Algorithm 1 The check of a form

Input: the key of the form; the row of the form in the certificate database of the reproduction pack, which gives G, s, τ_0, u_0; the rows of the form with the scope `general` in the tables `cert`, `ball`, `file` of the macadamia database

require for every stored file: SHA-256 of its bytes = the stored digest, size = the stored size (Algorithm 9)
for the row of kind `gsimple` of table `ball`, if any **do**
    require Algorithm 6 accepts the simple ball, with $P_{\mathrm{rec}} = \frac{4}{s}G$, τ_0, u_0 of the pack; the radius of the row = the radius of the certificate
end for
for the row of kind `gsharp` of table `ball`, if any **do**
    require Algorithm 7 accepts the ball, with $P_{\mathrm{rec}} = \frac{4}{s}G$, τ_0, u_0 of the pack; the radius of the row = the radius of the certificate
end for
read no row of the scope `fixedN` ▷ fixed-N certificates and balls: not used
build the registry of balls (Algorithm 4)
require Algorithm 2 accepts the tree, read from the stored tree file with the stored setup
require the numbers of leaves of each kind = the field `tree_kinds` of the row of table `cert`; the SHA-256 of the tree file = its field `tree_sha256`
accept, and print the kind of the certificate (Definition 26)

Algorithm 2 The tree walk (`verify_box_certificate` with the basin branches)

Input: G, s; the tree certificate (head and tree); the registry of balls

check the head as in [6, Algorithm 8]: $|\det U| = 1$, unit 4096, $G' = UGU^{\mathsf{T}}$, $C = \operatorname{Min} G'$ (exact), pool vectors nonzero, `"box_basis"` unimodular, L_{box} lower triangular with ones on the diagonal, triangles true relations, symmetry data verified ([6, Algorithm 6]); $R_0 \leftarrow$ the root box
stack $\leftarrow [(\text{tree}, R_0)]$; $n \leftarrow 0$
while stack not empty **do**
    (node, $[\ell, h]$) $\leftarrow$ pop
    if node is a leaf **then** $n \leftarrow n + 1$
        if kind `gball` or `basin` **then require** Algorithm 3 accepts the leaf on $[\ell, h]$ ▷ the added branch
        else if kind `outside` **then require** the combination of maps is positive on $[\ell, h]$
        else if kind `three` **then require** value ≥ 972 ▷ $\Delta \geq 972$
        else require value $\geq d_{\mathrm{cut}}$ (`box`, `pattern`, `pspec`, `uspec`) ▷ $d \geq d_{\mathrm{cut}}$
        end if
    else if node $= \{\text{v: } j,\ \text{at: } x,\ \text{le},\ \text{ge}\}$ with $\ell_j < x < h_j$ **then** push (le, $[\ell, h|_{h_j = x}]$) and (ge, $[\ell|_{\ell_j = x}, h]$)
    else reject ▷ an unfinished node
    end if
end while
accept iff n equals the number of leaves recorded in the head, when the head records one

Algorithm 3 The basin-leaf test (`gball_leaf_exact`, `box_in_some_ball`)

Input: the box $[\ell, h]$ (integers), $M_i L_{\mathrm{int}}$ and UU from the head; the leaf $\{\mathrm{gc}, \mathrm{gk}, \mathrm{gr2}\}$ (or $\{\mathrm{bc}, \mathrm{bk}, \mathrm{br2}\}$); the registry of general balls: centres $c_0, c_1, \ldots$ (rational, in $[0, 1)^9$), certified squared radii $r_0^2, r_1^2, \ldots$, the matrix P

function InBall(i, k, r'^2)
  require $0 \le i <$ number of centres, $k \in \mathbb{Z}^9$, $r'^2 \le r_i^2$
  for each of the 512 vertices t of $[\ell, h]$ **do**
    $\tau \leftarrow M_i L_{\mathrm{int}} t / UU$ ▷ exact rationals, eq. (4)
    require $P[\tau - c_i - k] \le r'^2$
  end for
  return true
end function
accept if InBall(gc, gk, gr2)
for the eight centres i nearest to the box **do** ▷ a saved index may point to another ball
  $k \leftarrow$ the integer vector nearest to (centre of the box) $- c_i$ ▷ a guess; the test is exact
  accept if InBall(i, k, r_i^2)
end for
reject

Algorithm 4 The registry of balls of a form

Input: G, s, τ_0 (pack); the setup of the tree (basis U, symmetry matrices); the simple ball (τ_0, r) accepted by Algorithm 6, or the ball (τ_0, r) accepted by Algorithm 7, if any

$P \leftarrow U P_{\mathrm{rec}} U^{\mathsf{T}} = \frac{4}{s} U G U^{\mathsf{T}}$; convert τ_0 and every τ_c to the basis U ($\tau \mapsto U^{-\mathsf{T}}\tau$)

$\mathcal{G} \leftarrow \{I, -I\} \cup$ the symmetry matrices V of the setup, each replaced by V or V^{T}, whichever satisfies $g^{\mathsf{T}} P g = P$ ▷ exact; a matrix that satisfies neither is not used

require $|\det g| = 1$ for every $g \in \mathcal{G}$

centres $\leftarrow$ the orbit of τ_0 modulo $\mathbb{Z}^9$ under the group generated by $\mathcal{G}$, the images $g\tau_0$ first, each with r^2 ▷ the ball and its images

key the registry by the set of minimal vectors of P

Algorithm 5 The check of a Delaunay subdivision

Input: the integer Gram matrix A (rank 10); the cells (c, R^2, V), $c \in \mathbb{Q}^{10}$, $R^2 \in \mathbb{Q}$, $V \subset \mathbb{Z}^{10}$

keys $\leftarrow \{\mathrm{canon}(c) \mapsto V\}$ ▷ canonical form modulo $\mathbb{Z}^{10}$ and $x \mapsto -x$

for every cell (c, R^2, V) **do**

  $X \leftarrow \{x \in \mathbb{Z}^{10} : A[x-c] \le R^2\}$ by exact enumeration

  require no $x \in X$ with $A[x-c] < R^2$ ▷ the sphere is empty

  require $\{x \in X : A[x-c] = R^2\} = V$ ▷ the vertex set is complete

  $F \leftarrow$ the facets of conv V (exact double description); **require** the cell is full-dimensional

  for every facet $f \in F$ **do**

    $(c', V') \leftarrow$ the neighbour across f: move the centre on the line of points equidistant from the vertices of f until the sphere meets a lattice point beyond f

    require $\mathrm{canon}(c') \in$ keys and keys$[\mathrm{canon}(c')] = \mathrm{canon}(V')$

  end for

end for

return $\rho^2 = \max R^2 / D_A$

Algorithm 6 The check of a simple ball (`l12general.verify_simple`; with a certificate of degree three for K, `l12general2.verify_simple`)

Input: $P = P_{\text{rec}}$, τ_0, u_0 (from the pack); the files of the simple ball: certificate (r, frame, proposals), reference data (cells, vertex constants, deep data, K, a), data of the contact cone of P

require the certificate's P, τ_0, u_0 = those of the pack and those of the reference data ▷ binding, Algorithm 1
require $r > 0$ ▷ $U(r)$ is a bound only for $r \geq 0$
$G_0 \leftarrow G(\tau_0, u_0)$; **require** $D_A G_0$ integral and $|\det U_{\text{red}}| = 1$; $A \leftarrow U_{\text{red}}(D_A G_0) U_{\text{red}}^{\mathsf{T}}$
require $\min A = 4D_A$ and the stored contacts $= \operatorname{Min} A$ ▷ exact enumeration
require $G_0^{-1} = \sum_c \varrho_c c c^{\mathsf{T}}$ with $\varrho \geq 0$ ▷ eutaxy, exact identity
require the certificate of K (Lemma 9 `b` or Lemma 10); $(a, b) \leftarrow$ from K; if a certificate of a is stored (Lemma 11), take the smaller a ▷ whole cone: no relative constants
require $\min P = 4$ and $d_0 = \det P / u_0$
require Algorithm 5 accepts the cells of A; the cell volumes add up to 1 (floating point; can only reject)
for every cell D **do**
    $\mu_D \leftarrow \max_v \left(\tilde{\mu}_v + \overline{\sqrt{K \operatorname{tr}((G_0 R_{m,v})^2)}}\right)$ if every vertex v of D has a stored certificate, else $b\, R_D^2$
    deep data of $D \leftarrow$ the stored $(\pi, c_D, R_{0,D})$ if all their checks pass, else none
end for
require the stored contacts of $P = \operatorname{Min} P$; $P^{-1} = \sum_c \varrho_c^P c c^{\mathsf{T}}$, $\varrho^P \geq 0$; the certificate of K_P; $(a_P, b_P) \leftarrow$ from K_P ($n = 9$), and a certified smaller a_P if stored
require every frame vector z_j is a list of exactly 9 integers, not all zero, and every $q_j \geq 0$
$U \leftarrow U(r)$ of Lemma 15 from the stored frame
$T^* \leftarrow$ the least bound of Lemma 16 that holds, from the proposed s_1, t_L, t_1 ▷ exact tests
require T^* defined, $aT^* < 1$, and the conditions of Theorem 19 at both ends of every piece of $[0, T^*]$
accept

Algorithm 7 The check of a ball at a stored lattice without margin (`l12gsharp.verify_ball`)

Input: $P = P_{\mathrm{rec}}$, τ_0, u_0 (from the pack); the certificate `CASE.gsharp.json` (r, frame, proposals), its cone file (contacts, generators, the identities of K and of a) and its cell file, each with its SHA-256 recorded in the certificate

require the file is well formed: $r > 0$, every frame vector a list of exactly 9 integers, every weight $q_j \geq 0$
require the certificate's P, τ_0, u_0 = those of the pack; P positive definite, $\min P = 4$ ▷ binding
require the SHA-256 of the cone file and of the cell file = those recorded
$G_0 \leftarrow G(\tau_0, u_0)$; **require** $A = U_{\mathrm{red}} G_0 U_{\mathrm{red}}^{\mathsf{T}}$ integral, $|\det U_{\mathrm{red}}| = 1$, $\min A = 4$, $d_0 = \det P / u_0$
require the contacts have norm 4; every generator is an integer automorphism of A
require the identity (5) for K (Lemma 10), expanded over the orbits, holds exactly
require the identity of Lemma 11 for a holds exactly, with zero residual
require $G_0^{-1} = \sum_c \varrho_c c c^{\mathsf{T}}$ with $\varrho_c = 4 \sum_d w_{cd}/(1 + 10a) \geq 0$ ▷ eutaxy, exact identity
require $d_0\big(4 - \max_D R_D^2\big) \geq 972$ ▷ the centre may have no margin
require Algorithm 5 accepts the cells of A; the cell volumes add up to 1 (floating point; can only reject)
$U \leftarrow U(r)$ of Lemma 15 from the stored frame; **require** $U > 0$
$T^* \leftarrow t_L$ by Lemma 21 and Lemma 16 `b`, from the proposed $\bar{\sigma}$, β, t_L ▷ exact tests
require T^* defined and the test of Theorem 23 at $T = T^*$ passes for every cell (the cell test of Section 3.4)
accept

Algorithm 8 Storing a general certificate (`add-general`)

Input: the output folder of the run for one form; its key; the build `ZIP` of the checkers

1: **require** the form has no general certificate yet ▷ append-only
2: **require** the entry of the form in `tree_results.json` has `"general_mode"` and `"tree_valid"` true and the status `excluded`; no leaf of kind `cwit` ▷ a leaf valid for $N = P$ only
3: **require** `"tree_sha256"` = SHA-256 of the tree file; the census of the leaves of the tree file, read again by the program, = the leaf kinds of the entry; every leaf `gball` holds integers only
4: **for** every ball file of the folder **do**
5: **require** r is a pair of positive integers; the frame is absent or has vectors of exactly 9 integers, not all zero, and weights ≥ 0; the record is valid, made with the cells checked, by a checker allowed for the theorem, with the radius of the file and the SHA-256 of the stored bytes; for a ball `gsharp`, the SHA-256 of its data files = those of the certificate
6: **end for**
7: **require** all balls are about the same P; the build holds every module the certificate needs; no path leaves the folder
8: in one transaction: insert the rows `file`, `ball` and `cert`; read them back before the commit

Algorithm 9 Exporting a form (`export CASE DIR --scope general`)

Input: the key of the form; a folder

1: **for** every row of table `file` of the general certificate of the form **do**
2: decompress `blob` if its codec is `zlib`; **require** size and SHA-256 = the stored ones ▷ a mismatch stops the export
3: write it to `DIR/RELPATH`; **require** SHA-256 of the written file = `sha256`
4: **end for**