Two active-set methods for smallest enclosing balls

Thomas Schmelzer

September 7, 2026

Abstract

Two active-set methods solve the smallest enclosing ball problem from opposite sides: the dual ascent over the unit simplex of Dearing and Zeck [3], whose ball encloses nothing until it terminates, and the primal deflation of Fischer, Gärtner and Kutz [5], which is feasible at every step. This note compares the two with their linear algebra held fixed. They keep the same support set, solve the same subproblem, and can repair the same QR factorisation of the same edge matrix in $O(dr)$; handing both that one data structure removes the usual confound, so what separates them is a property of the two searches rather than of two implementations. On Gaussian clouds to $d = 1.6 \cdot 10^4$ they agree on the ball and on its support on every row, and the primal method takes 1.2 to 2.1 times as many steps — the price of never taking a full step — which the wall clock damps to a factor of 1.1 to 1.6. The cone program both replace, handed to an interior-point solver, is two to three orders of magnitude slower where it runs at all, and stops at a tolerance where both of these terminate exactly. Neither method is ours; both ship in `cvxball`.

1 Introduction

Sylvester stated the problem in 1857 in a single sentence [8]: given a finite set of points in the plane, find the smallest circle containing all of them. It is the 1-centre facility location problem; it is the primitive under bounding-sphere hierarchies in graphics and collision detection; and as the minimum enclosing ball of a set of feature vectors it is the core-set problem behind kernel machines.

The algorithmic history runs in two lines that have largely stayed apart. The first is computational-geometric. Welzl's randomised incremental method [9] solves the problem in expected $O(n)$ time for fixed d, with a constant that grows superexponentially in d; Gärtner [6] made its predicates robust in floating point, Gärtner and Schönherr [7] recast the problem as a quadratic program and solved it exactly, and Fischer, Gärtner and Kutz [5] left the recursion behind for a pivoting method that deflates an enclosing ball, carrying a dynamic QR of the support's edge matrix and reporting d in the thousands — a finite algorithm of that second kind going back to Elzinga and Hearn [4]. The second line is an optimization one, reaching the same object from the dual: Dearing and Zeck [3] gave a dual algorithm maintaining a support set and raising the radius to R from below, and Cavaleiro and Alizadeh brought its iteration from $O(d^3)$ to $O(d^2)$ by updating a factorisation rather than rebuilding it [1] and then extended the method to balls [2].

This note states the two methods in a common notation (Sections 2–4), shows that they can share one data structure (Section 5), and reports what separates them once that data structure is held fixed (Section 6). Handing both the same linear algebra removes the usual confound: what is left is a property of the two searches rather than of two implementations. Neither method is new and none is claimed here; what this note adds is the controlled comparison, and an implementation of both that makes it reproducible.

2 The problem and its two sides

For $S = \{p_1, \dots, p_n\} \subset \mathbb{R}^d$ the smallest enclosing ball is the solution of

$$\min_{x,r} \; r \quad \text{s.t.} \quad \|p_i - x\| \le r, \; i = 1, \dots, n, \tag{1}$$

a second-order cone program in $(r, x) \in \mathbb{R}^{1+d}$ with n blocks $(r,\, p_i - x) \in \mathcal{Q}^{d+1}$. Existence follows from coercivity of $f(x) = \max_i \|p_i - x\|$ and uniqueness from strict convexity of f^2; we write $(R, x^\star)$ for the optimum.

Let $\Delta_n = \{u \in \mathbb{R}^n : u \ge 0, \; \mathbf{1}^\mathsf{T} u = 1\}$, and for $u \in \Delta_n$ put $x(u) = \sum_i u_i p_i$ and

$$\varphi(u) = \sum_i u_i \|p_i\|^2 - \|x(u)\|^2, \tag{2}$$

a concave quadratic. Everything below rests on the parallel-axis identity, valid for all $y \in \mathbb{R}^d$ and $u \in \Delta_n$:

$$\sum_i u_i \|p_i - y\|^2 \;=\; \varphi(u) + \|x(u) - y\|^2, \tag{3}$$

obtained by expanding and collecting, using $\sum_i u_i p_i = x(u)$ on the cross term and $\mathbf{1}^\mathsf{T} u = 1$ on $\|y\|^2$. Measuring from anywhere other than the weighted mean costs precisely $\|x(u) - y\|^2$, and $\varphi(u)$ is what is left when y is that mean.

Proposition 1. *$R^2 = \max_{u\in\Delta_n} \varphi(u)$, and $x^\star = x(w)$ for any maximiser w.*

Both directions are short: $\varphi \le R^2$ is (3) applied to any enclosing ball, and equality follows by separating $x^\star$ from $\operatorname{conv} T$, $T = \{p_i : \|p_i - x^\star\| = R\}$, if the two were disjoint. See also [7]. Since $\partial\varphi/\partial u_i = \|p_i - x(u)\|^2 - \|x(u)\|^2$, the first-order conditions read, after cancelling the common term,

$$\|p_i - x\| = R \;\; (w_i > 0), \qquad \|p_i - x\| \le R \;\; (w_i = 0), \tag{4}$$

a geometric certificate: the points carrying weight lie on the boundary sphere and the rest are enclosed, which a caller checks in one pass. By Carathéodory the support may be taken of size at most $d+1$, so (1) is determined by at most $d+1$ of the n points and the problem is combinatorial in the choice of that subset.

Read forwards, (4) is a search for weights: keep a set of points on the sphere, adjust the weights, stop when no point lies outside. Read backwards it is a search for balls: keep a ball containing S with a set T on its boundary, and shrink it until its centre falls inside $\operatorname{conv} T$. The second rests on a fact due to Seidel, recorded in [7] and Lemma 1 of [5]: if T lies on the boundary of a ball B centred at c, then B is the smallest ball enclosing T if and only if $c \in \operatorname{conv} T$. With $S \subseteq B$ and $T \subseteq S$ the inclusions $\operatorname{seb}(T) \subseteq \operatorname{seb}(S) \subseteq B$ close, so $\operatorname{conv} T$ is the only thing left to check.

Sections 3 and 4 are those two readings made into methods. They keep the same kind of state — a subset of S that will end up on the boundary, and a point in its affine hull — and differ in which feasibility they refuse to give up. The first stays dual feasible, so by Proposition 1 its radius rises to R from below and its ball encloses nothing until the last iteration; the second stays primal feasible, so its radius falls to R from above and every iterate is an answer, merely not yet the best one.

3 The dual active-set method

This is the dual algorithm of Dearing and Zeck [3] as sharpened by Cavaleiro and Alizadeh [1, 2], restated so that the correspondence with Section 4 is exact. Two details below depart from that line of work and are given as differences rather than as claims: the step rule here is a full step to the circumcentre of the current face followed by a ratio test (6), where [2] performs an exact curve search; and an affinely dependent support is resolved by a descent direction in the null space rather than by dropping a point before the search begins.

Maintain a working set F and weights $u \in \Delta_n$ supported on F with $\varphi(u)$ non-decreasing. This is a primal active-set method run on the dual (2): the free variables are the weights of the support, every subproblem solves one face exactly, and feasibility for (1) arrives only at the end.

Subproblem. Maximising φ over weights supported on F with no sign restriction is, by (4) without the inequalities, the requirement that x be equidistant from $\{q_j\}_{j\in F}$ and lie in their affine hull. With D the $(m-1)\times d$ matrix of edges $e_j = q_j - q_0$ and $x = q_0 + D^{\mathsf T} y$,

$$(DD^{\mathsf T})\, y = \tfrac{1}{2}\big(\|e_1\|^2, \dots, \|e_{m-1}\|^2\big)^{\mathsf T}, \qquad w = \Big(1 - \textstyle\sum_j y_j,\; y\Big), \tag{5}$$

a dense system of order $m-1 \le d$, non-singular exactly when F is affinely independent. This is the circumcentre of the face in barycentric coordinates.

Dropping. If $\min_j w_j < 0$ the circumcentre lies outside $\operatorname{conv}\{q_j\}$. Move along $s = w - u$ by the ratio test

$$\alpha = \min\{-u_i/s_i : \; s_i < 0\}, \tag{6}$$

which drives at least one weight to zero; remove it from F.

Degeneracy. If F is affinely dependent, $DD^{\mathsf T}$ is singular. The directions s with $\mathbf{1}^{\mathsf T} s = 0$ and $D^{\mathsf T} s_{1:} = 0$ leave $x(u)$ fixed, so $\varphi(u + ts) = \varphi(u) + t\sum_i s_i \|p_i\|^2$ is linear and the subproblem is unbounded. Project the gradient onto that null space and apply (6), shrinking F by one.

Adding. Otherwise accept $u \leftarrow w$, set $x = x(u)$ and $R = \max_{i\in F}\|p_i - x\|$, and let $k = \arg\max_i \|p_i - x\|$. If $\|p_k - x\| \le R$ then (4) holds and (R, x, u) is optimal; otherwise append k to F with weight zero. Since $\partial\varphi/\partial u_k - \partial\varphi/\partial u_i = \|p_k - x\|^2 - \|p_i - x\|^2$, the farthest violator is the fixed variable with the largest reduced gradient, so this is also the steepest-ascent choice.

Each iteration either strictly increases φ or strictly shrinks F, and there are finitely many working sets, so the method is finite; a degenerate zero-length step needs an anti-cycling rule in exact arithmetic, and in practice an iteration cap serves as the safety net.

4 The primal pivoting method

The method of Fischer, Gärtner and Kutz [5] carries a ball rather than a set of weights, and never surrenders it. Its state is a pair (c, T), a centre and a support set, subject to the invariant of their Fig. 2:

$$S \subseteq B(c,T), \qquad T \subseteq \partial B(c,T), \qquad T \text{ affinely independent}, \tag{7}$$

where $B(c,T)$ is the ball about c through the farthest point of T. By Seidel's lemma the pair is optimal as soon as $c \in \operatorname{conv} T$. Each iteration is a dropping phase, entered only once c has landed in $\operatorname{aff}(T)$, followed by a walking phase.

Dropping. $c \in \operatorname{aff}(T) \setminus \operatorname{conv} T$ means some affine coefficient of c with respect to T is negative; drop such a point. The coefficients are the same object as the barycentric weights of (5) and come from the same factorisation (Section 5), but here they are read rather than moved: there is no ratio test, because c does not travel during a drop. Their Lemmata 4 and 5 together say the dropped point cannot stop the walk that immediately follows.

Walking. Move c along $e = \operatorname{cc}(T) - c$. Their Lemma 3 says that segment is orthogonal to $\operatorname{aff}(T)$, that T stays on the boundary the whole way, and that the radius strictly decreases. Two consequences make the step cheap. Orthogonality makes $\operatorname{cc}(T)$ the projection of c onto $\operatorname{aff}(T)$, so no circumsphere is ever solved for and one projection does the work of (5). And the stopping fraction has a closed form: with $c(a) = c + ae$, a point p joins the boundary when $\|p - c(a)\|$ equals the common support distance; the a^2 terms cancel, and $\langle q - c, e\rangle = \|e\|^2$ for every $q \in T$ leaves

$$a_p = \frac{R_c^2 - \|p-c\|^2}{2\big(\|e\|^2 - \langle p-c,\, e\rangle\big)}, \qquad R_c = \max_{q\in T}\|q-c\|, \qquad a = \min\{1,\ \min_p a_p\}. \tag{8}$$

Criterion (1) of Lemma 4 — that p lies behind $\operatorname{aff}(T)$ and so cannot stop the walk — is exactly the statement that this denominator is non-positive, so one sign test separates the candidates. If $a = 1$ the centre arrives at $\operatorname{cc}(T)$, lands in $\operatorname{aff}(T)$, and the next iteration drops; otherwise the minimising point enters T and the ball has shrunk.

Pivot rules. Ties are resolved two ways. Bland's rule, adapted — fix an order on S, take the smallest-rank candidate at both the drop and the admission — is what their Theorem 1 proves terminating, degeneracies included, and is also recorded there as slow. The heuristic, which their own code runs and which Table 2 reports, drops the *most* negative coefficient and admits the tied point farthest from $\operatorname{aff}(T)$, leaving the support best conditioned. The radius falls strictly on every walk and no configuration recurs, so the method is finite.

How the two differ. The support sets agree and the drop criteria agree — a negative barycentric coordinate, in both — and the mechanics agree nowhere else. Section 3 takes the whole step to $\operatorname{cc}(F)$ and only then consults the data, admitting the farthest violator; this method never takes a full step while any point is still outside, truncating the walk at whichever point first touches the shrinking sphere, which is also the point that enters. A drop here is immediate and followed by a resumed walk; there it is a ratio test in weight space, moving every weight at once. The invariant (7) also requires T affinely independent, with no fallback if it is not, so an entering point must be certified off $\operatorname{aff}(T)$ before admission; Section 3 instead permits a dependent support and answers it with a null-space direction. Neither choice dominates — one avoids the case, the other resolves it.

5 The shared factorisation

Per iteration, both methods want the same two things from their support: whether it is affinely independent, and where the circumcentre of its face lies. Both read those off the same object. Let D be the $r \times d$ matrix of edges $q_j - q_0$ of the current support and let $D^{\mathsf{T}} = QR$ be its economic factorisation. Three identities take every per-iteration quantity off d and onto the $r \times r$ block:

$$DD^{\mathsf{T}} = R^{\mathsf{T}}R, \qquad \ker D^{\mathsf{T}} = \ker R, \qquad \|q_j - q_0\|^2 = \|Re_j\|^2. \tag{9}$$

The first makes (5) two triangular solves against a Cholesky factor already to hand, with no $O(r^2 d)$ product; the second makes the affine-dependence test a rank test on R — note it is the *right* null space, not the one the edge matrix itself would hand you; the third supplies the right-hand side. The pivoting method reads the same R for its projection and its coefficients.

Rebuilding that factorisation costs $O(dr^2)$ and repairing it costs $O(dr)$. The repair is section 4 of [5], and the same move on the dual side is [1]. Three updates cover every move either method makes: a point joining the support appends a column, a point other than q_0 leaving deletes one, and q_0 leaving has no column of its own — every column being measured *from* it — so q_1 is promoted by a deletion followed by a rank-one update. In a compiled language these are Givens sweeps; in an interpreted one they have to be the library's, or a single vectorised refactorisation will beat them and the measurement will be of the interpreter rather than of the data structure.

Two things keep this from being a pure win. The dependent supports Section 3 permits are exactly what an economic QR cannot hold — an update refuses a column already in the span — so those faces are met by refactorising. And the repair is not cheaper at every size: on the clouds of Section 6 it is worth a factor of 3.5 at $d = 8\,000$, is at parity near $d = 100$, and loses below that, so an implementation dispatches on d rather than committing to one route.

6 A numerical comparison

Table 2 fixes $n = 10^3$ and takes d from 10^3 to $1.6 \cdot 10^4$, the regime [5] was written for and the first in which the linear algebra rather than the combinatorics decides the cost. Two quantities are reported, because either alone would mislead. The first is the relative enclosure error $\max_i \|p_i - x\|/R - 1$, a deterministic property of the returned pair, free of timing noise, where a positive value means the ball does not in fact contain the cloud. The second is wall-clock time, which must be read as a comparison of *implementations*: both methods spend most of each iteration in one vectorised sweep and one repair of the factorisation, so the step counts beside them are what carry over to another language.

Before that, Table 1 says what the cone program (1) costs when handed to a solver directly, since a comparison against two active-set methods is only informative if the obvious alternative is priced. Clarabel is given the same clouds and the program assembled by hand — n second-order cones of dimension $d + 1$, no modelling layer in between.

	enclosure error			time (s)			
d	act. set	pivoting	Clarabel	act. set	pivoting	Clarabel	ratio
62	0	0	$-1.4 \cdot 10^{-11}$	0.003	0.006	0.5	165
125	0	0	$-9.1 \cdot 10^{-10}$	0.005	0.006	1.1	231
250	0	0	$8.4 \cdot 10^{-12}$	0.026	0.032	98.7	3789
500	0	0	$-5.8 \cdot 10^{-10}$	0.056	0.058	193.5	3433
1 000	$-4.4 \cdot 10^{-16}$	$-2.2 \cdot 10^{-16}$	$-1.9 \cdot 10^{-10}$	0.111	0.167	359.6	3251

Table 1: The cone-program reference at $n = 10^3$: one standard normal cloud per row, the same cloud given to all three methods. *Ratio* is Clarabel's time over the active-set method's. The two active-set methods are timed as the median of three runs; Clarabel once per row above $d = 125$, where a single solve already runs to minutes. One run is one draw, and Clarabel's cost is the product of a per-iteration solve that grows with d and an iteration count that does not, so these times carry the spread of that second factor and no trend should be read off five of them.

Two things separate the reference from the two methods this note is about, and only one of them is speed. Clarabel is two to three orders of magnitude slower on every row — it factors a KKT system built from $n(d+1)$ rows, where neither active-set method ever factorises anything of size n. But the errors differ in *kind*. An interior-point method stops at a tolerance, so its ball is right to a few parts in 10^9 and, at $d = 250$, is very slightly wrong in the direction that matters: the enclosure error is positive, so the returned ball does not quite contain the cloud. Both active-set methods terminate at a vertex and are exact to the last bit on every row. That difference is structural rather than a matter of tuning, and it is the reason the shipped solvers are not wrappers around a cone solver.

The step counts separate by about a factor of two, in the direction the mechanics predict. The active-set count equals the final support size on four of the five rows — on Gaussian data the method adds points and essentially never drops one — while the pivoting method takes 1.2 to 2.1 times as many steps, the price of never taking a full step: each walk is truncated by the first point of S to touch the shrinking sphere. The wall clocks do not separate by anything like that factor. Their ratio sits between 1.1 and 1.6 with no trend in d, and repeat runs move it by a few percent, so it is a small constant rather than a curve: with r in the hundreds both methods spend most of each iteration in the same two places, and the pivoting method's extra steps are its cheapest ones. Since both columns were measured with the factorisation repaired rather than rebuilt, the ratio between them measures what the two searches cost and not what two data structures cost.

	steps			enclosure error		time (s)		
d	support	act. set	pivoting	act. set	pivoting	act. set	pivoting	ratio
1 000	92	92	197	$-4.4 \cdot 10^{-16}$	0	0.148	0.232	1.56
2 000	124	128	208	$-1.1 \cdot 10^{-16}$	$-1.1 \cdot 10^{-16}$	0.302	0.355	1.18
4 000	177	177	249	$-6.7 \cdot 10^{-16}$	0	1.296	1.450	1.12
8 000	217	217	328	$-1.0 \cdot 10^{-15}$	0	2.760	3.272	1.19
16 000	283	283	352	$-1.4 \cdot 10^{-15}$	0	7.869	8.457	1.07

Table 2: Growth in d at $n = 10^3$: standard normal clouds, one per row, median of three runs. Apple M4 Pro, CPython 3.12.13, NumPy 2.5.1, SciPy 1.18.0. Both methods repair the QR of their support's edge matrix rather than rebuilding it, and both identify the same support set, whose size is the first column. *Steps* are active-set iterations against pivot steps, which count comparable work here: one solve or projection of order r plus one $O(nd)$ sweep each. Neither reference is run here: Welzl's median basis count is already $2.8 \cdot 10^5$ at $d = 11$, and Clarabel — already minutes per solve at $d = 10^3$ in Table 1 — would be handed a program of $n(d+1) \approx 1.6 \cdot 10^7$ rows to factor.

Two further things are worth naming. The support grows like $\sqrt{d}$ — 92 points at $d = 10^3$ and 283 at $d = 1.6 \cdot 10^4$ — staying an order of magnitude below the Carathéodory bound $d+1$, so the per-iteration solve is of that order rather than of d; with $O(\sqrt{d})$ iterations of $O(nd)$ each the total should scale as $d^{3/2}$, and it does, a 53-fold increase in time over a 16-fold increase in d. And the enclosure errors stay at a few ulp for both methods across four doublings of d: the exactness is a property of terminating at a basis, not of small d.

7 Which to use

Per iteration each method performs one $O(nd)$ sweep of the cloud — squared distances to the centre in Section 3, the stopping fractions (8) in Section 4 — and one solve or projection of order at most d. Nothing of size n is ever factorised, which is the structural difference from an interior-point method applied to (1), and both solve the final face exactly rather than leaving the caller to decide which slacks of size 10^{-9} were meant to be zero.

What they can hand back differs, and that is the practical reason to keep both. The pivoting method is primal feasible at every step: interrupt it at any iteration and what is on hand is an enclosing ball, correct if not yet smallest, with a radius that only falls from here. That is the guarantee to want under a time budget, and the dual method cannot offer it — by Proposition 1 its radius is a lower bound throughout. In exchange the dual method finishes with the weights, which are the conic dual of (1) after one rescaling: $z_i = u_i(1, -(p_i - x)/R)$ lies in $\mathcal{Q}^{d+1}$, is tight exactly on the support, and pairs with the slack $s_i = (R, p_i - x)$ to give $s_i^{\mathsf{T}} z_i = 0$. A caller can verify optimality from (4) in one pass over the data, without re-solving. It is also the faster of the two on every cloud measured, which is why `cvxball` makes it the default.

Both methods are available at `https://github.com/tschm/cvxball`, sharing the factorisation of Section 5, with the Clarabel cone program and Welzl's recursion beside them as references.

References

[1] M. Cavaleiro and F. Alizadeh. A faster dual algorithm for the Euclidean minimum covering ball problem. *Annals of Operations Research*, 2018. DOI 10.1007/s10479-018-3123-5.

[2] M. Cavaleiro and F. Alizadeh. A dual simplex-type algorithm for the smallest enclosing ball of balls. *Computational Optimization and Applications* 79(3):767–787, 2021.

[3] P. M. Dearing and C. R. Zeck. A dual algorithm for the minimum covering ball problem in $\mathbb{R}^n$. *Operations Research Letters* 37(3):171–175, 2009.

[4] J. Elzinga and D. W. Hearn. Geometrical solutions for some minimax location problems. *Transportation Science* 6(4):379–394, 1972.

[5] K. Fischer, B. Gärtner and M. Kutz. Fast smallest-enclosing-ball computation in high dimensions. *Proc. ESA*, 630–641, 2003.

[6] B. Gärtner. Fast and robust smallest enclosing balls. *Proc. ESA*, 325–338, 1999.

[7] B. Gärtner and S. Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, 110–118, 2000.

[8] J. J. Sylvester. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics* 1:79, 1857.

[9] E. Welzl. Smallest enclosing disks (balls and ellipsoids). *New Results and New Trends in Computer Science*, LNCS 555, 359–370, 1991.