Student Research Symposium | Student Research Symposium 2015

May 12th, 11:00 AM - 1:00 PM

The Smallest Intersecting Ball Problem

Daniel J. Giles
Portland State University

Mau Nam Nguyen
Portland State University

Giles, Daniel J. and Nguyen, Mau Nam, "The Smallest Intersecting Ball Problem" (2015). *Student Research Symposium*. 15.
https://pdxscholar.library.pdx.edu/studentsymposium/2015/Posters/15

The Smallest Intersecting Ball Problem

Daniel Giles
Faculty Advisor: Dr Mau Nam Nguyen
Fariborz Maseeh Department of Mathematics and Statistics
PO Box 751, Portland, Oregon 97207

Portland State UNIVERSITY

Abstract

The smallest intersecting ball problem asks for the smallest radius necessary to intersect a collection of m closed sets. Formally, we write

$$\min_{x\in\mathbb{R}^n} \mathcal{D}(x) = \max\{\text{dist}(x,\Omega_i) | i = 1, 2, ..., m\}$$

This research explores various methods of finding the solution, and some tools of convex optimization which facilitate these methods. The max distance function is non-smooth and convex, which lends itself to minimization by the classical subgradient method. A second approach uses a log-exponential smoothing approximation of the max distance function, coupled with distance majorization and Nesterov acceleration. Two original algorithms are presented: The first method expands the sets and finds their smallest non-empty intersection, in which the optimal solution is proven to lie. The other–weighted projections– searches for the optimal solution as a convex combination of projections onto each set, with coefficients iteratively updated based on which set is most distant. Each algorithm is implemented in various scenarios, with the log-exponential smoothing algorithm exhibiting fastest convergence. The weighted projection method is consistently competitive, but its convergence remains unproven.

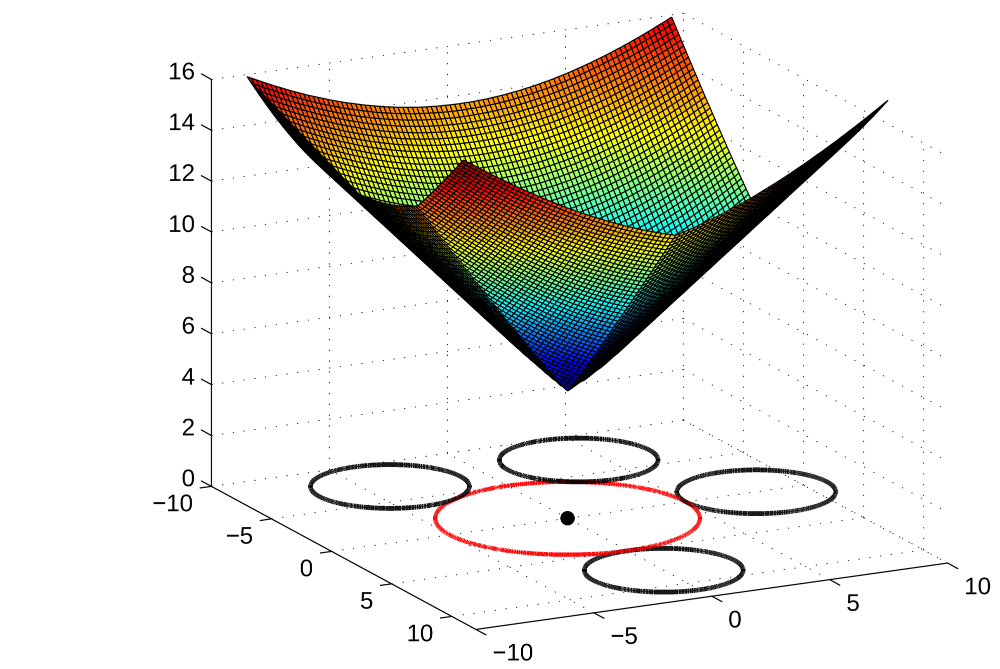


Figure 1: Max distance function evaluated in $\mathbb{R}^2$; the optimal ball is shown in red. The sharp corners of $\mathcal{D}(x)$ exist where two sets are equidistant. The subgradient vector may point toward either.

Subgradient Method

The subgradient method starts with an initial x_0, and each update is given by

$$x_{k+1} := x_k - \alpha_k g_k$$

where g_k is a subgradient at x_k and α_k is a scalar step size. For the max distance function, assuming x does not lie in its most distant set, we use the subgradient

$$g = \frac{x - P(x, \Omega_S)}{\|x - P(x, \Omega_S)\|}$$

where Ω_S satisfies $\text{dist}(x, \Omega_S) = \mathcal{D}(x)$, and $P(x, \Omega_S)$ is the Euclidean projection onto Ω_S. As $k \to \infty$, the subgradient method may not converge to the optimal solution, but a bound on the error may be computed as a function of step size.

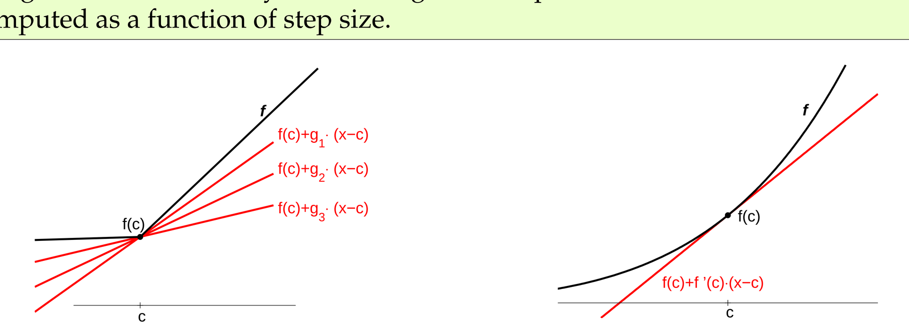


Figure 2: The subgradients (left) of f at c are all g such that $f(x) \geq f(c) + \langle g, x - c\rangle$. Compare to gradient-based linear approximation (right) of f at c, by $\mathcal{L}(x) = f(c) + \langle \nabla f(c), x - c\rangle$

Majorization Minimization Principle

The majorization-minimization (MM) principle finds a local minimum of a function f by minimizing a sequence of surrogate functions. The surrogate function $g(x|y)$ must satisfy the following two conditions:

Tangency:	$g(x\|x) = f(x) \quad \forall x \in \mathbb{R}^n$
Domination:	$g(x\|\cdot) \geq f(x) \quad \forall x \in \mathbb{R}^n$

Then from some x_0, we define x_{k+1} :=arg min $g(x|x_k)$, which provides a monotonically decreasing sequence:

$$\underbrace{f(x_n) = g(x_n|x_n)}_{\text{by tangency}} \geq \overbrace{\underbrace{g(x_{n+1}|x_n) \geq f(x_{n+1})}_{\text{by dominance}}}^{x_{n+1}:=\text{opt. sol } g(x|x_n)}$$

If f is differentiable and convex, it can be shown that this is indeed a strictly decreasing sequence except at the stationary points of f, to which x_k converges. For a surrogate to the distance function $\text{dist}(x, \Omega) = \|x - P(x, \Omega)\|$, we use $g(x|x_k) := \|x - P(x_k, \Omega)\|$. In this way, g approximates the distance to a set with instead the distance to some element in that set. From this we obtain an algorithm for locating an element in an intersection.

Feasible Point in an Intersection

Given m closed convex sets $C_1, ..., C_m$ in $\mathbb{R}^n$, we may employ the MM principle to find a point in their intersection $\bigcap C_i$, if it is non-empty. This is equivalent to finding a solution to

$$\min_x:\ f(x) = \frac{1}{2}\sum_i^m \|x - P_{C_i}(x)\|^2,$$

but an explicit solution is non-trivial. A majorizing surrogate to f is given by $g(x|x_k) := \frac{1}{2}\sum \|x - P_{C_i}(x_k)\|^2$, and for fixed x_k, solving for $\nabla g(x) = 0$ is straight-forward. Using this minimum to define our next iteration gives us the update:

$$x_{k+1} := \frac{1}{m}\sum_i^m P_{C_i}(x),$$

which is the average of projections onto each set. This will be used in the expanding sets algorithm to find a smallest intersecting ball.

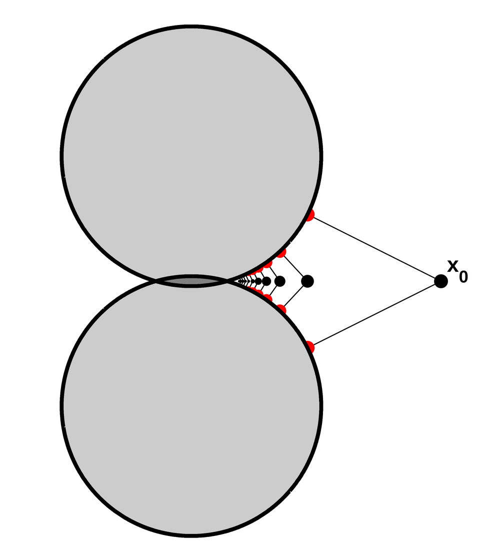


Figure 3: Each update of the feasible point algorithm minimizes the distance to projections (red) of the prior iteration onto a collection of sets.

Log Exponential Smoothing

The max distance function $\mathcal{D}(x)$ can be approximated by a log-exponential smoothing function

$$\mathcal{G}(x, p) = p \ln \sum_{i=1}^m e^{\frac{\sqrt{\text{dist}(x,\Omega_i)^2 + p^2}}{p}}$$

for some $p > 0$ with an error given by

$$0 \leq \mathcal{G}(x, p) - \mathcal{D}(x) \leq p(1 + \ln(m)).$$

To minimize this now differentiable function, we will apply the MM principle. The surrogate function utilizes distance majorization in the same way as the feasible point problem, with the surrogate given by:

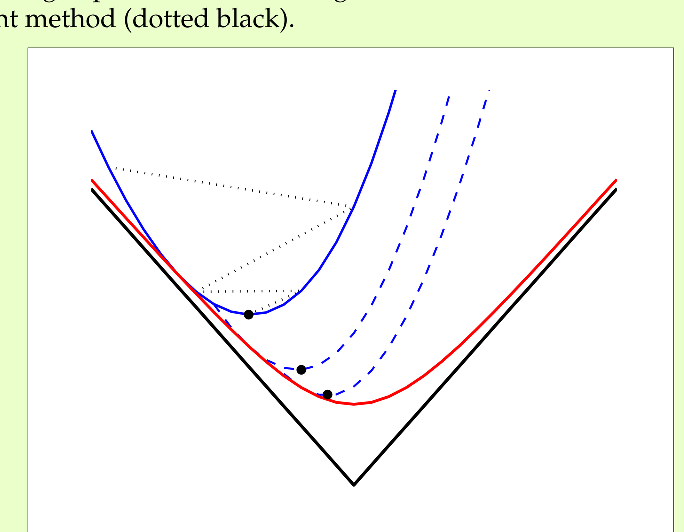
$$\mathcal{G}_p(x, x_k) := p \ln \sum_{i=1}^m e^{\frac{\sqrt{\|x - P(x_k,\Omega_i)\|^2 + p^2}}{p}},$$

but in this case an algebraic solution for x such that $\nabla\mathcal{G}(x, x_k) = 0$ is not available. So to minimize each iterate of the surrogate function, we use a form of Nesterov's accelerated gradient method to solve numerically, with updates given by (with $L = 2/p$ as the Lipschitz constant of $\nabla\mathcal{G}$)

$$x_{k+1} = \frac{2}{k+3}\left(x_k - \frac{1}{L}\sum_{i=0}^k \frac{i+1}{2}\nabla f(x_k)\right) + \frac{k+1}{k+3}\left(x_k - \frac{1}{L}\nabla f(x_k)\right)$$

A Minimization Model

The black line in this image represents the non-smooth max distance function, approximated by the (red) log-sum-exponential function for a given p. To find its optimal solution, we invoke the MM-principle with a sequence of surrogate functions (blue) whose optimal solutions approach that of the log-exponential. The surrogates are themselves minimized by Nesterov's accelerated gradient method (dotted black).



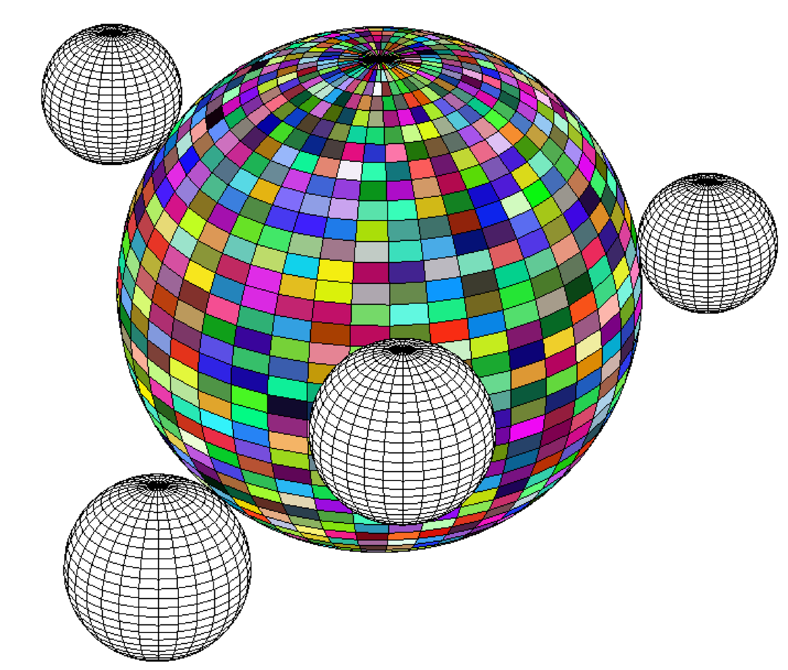


Figure 4: Smallest intersecting ball in $\mathbb{R}^3$.

Set Expansion

For any set Ω_i define its t-expansion $\Omega_{i,t}$ as

$$\Omega_{i,t} = \{x \in \mathbb{R}^n | \text{dist}(x, \Omega_i) \leq t\}$$

and let

$$T = \inf\{t \in \mathbb{R} | \bigcap_i \Omega_{i,t} \neq \varnothing\}$$

that is, T is the smallest expansion term such that the intersection of all expanded sets is non-empty.

Proposition Any element $x \in \bigcap \Omega_{i,T}$ is an optimal solution to $\mathcal{D}(x)$.

Proof. Choose some $y \in \cap\Omega_{i,T}$. By its inclusion in the intersection, $d(y, \Omega_i) \leq T$ for all i. This includes its most distant set, so $\mathcal{D}(y) \leq T$. But then $y \in \bigcap \Omega_{i,\mathcal{D}(y)}$, and with T being the infimum value for non-empty intersection, we also have $T \leq \mathcal{D}(y)$. Thus for any $y \in \cap\Omega_{i,T}$, we have $\mathcal{D}(y) = T$.

Now consider the optimal solution x^*. For all Ω_j, we have $d(x^*, \Omega_j) \leq \mathcal{D}(x^*) \leq \mathcal{D}(y) = T$. With the distance from x^* to any set bound by T, we have

$$x^* \in \bigcap_i \Omega_{i,T} \quad \text{from which it follows that} \quad \mathcal{D}(x^*) = T = \mathcal{D}(y)$$

So y is an optimal solution. □

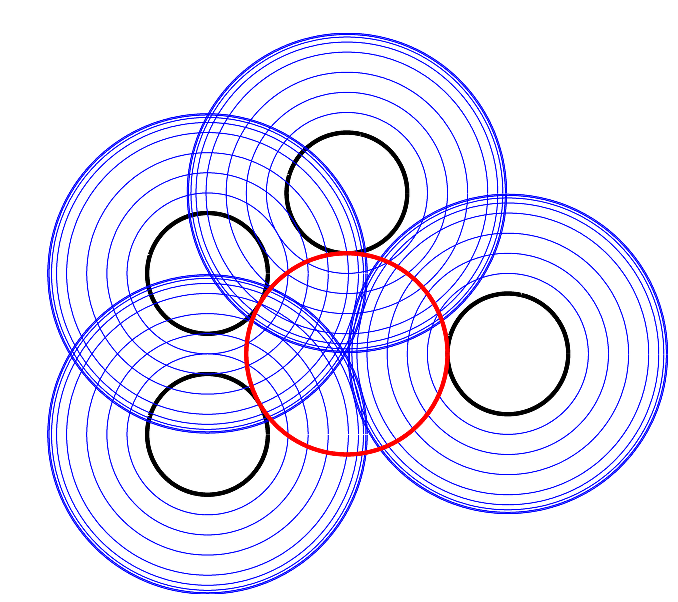


Figure 5: Expanding sets in $\mathbb{R}^2$. An element lies in the smallest non-empty intersection of expanded sets if and only if it is an optimal solution.

Finding the Smallest Non-Empty Intersection

It has been shown that the set $\bigcap_i \Omega_{i,T}$ contains an optimal solution, but still we must locate that set. If we expand the sets until their intersection is non-empty, the feasible point algorithm can locate a point in it (assuming the sets are convex.) But without infinitesimal steps, we overshoot the necessarily infimal expansion. So we shrink the sets back for an empty intersection, and proceed again with smaller steps. A full algorithm is given:

```
initialize x_0, t_0 = 0, τ > 0, σ ∈ (0,1), ε
repeat
    define all Ω_{i,t_k}
    use feasible point algorithm for ∩Ω_{i,t_k} until convergence at y
    if y ∈ ∩Ω_{i,t_k} then
        t_{k+1} := t_k − τ
        τ := στ
    else
        t_{k+1} := t_k + τ
    end if
    k := k + 1
    x_{k+1} := y
until max{dist(x_k, Ω_{i,t_k})} ≤ ε
```

Weighted Projections *in progress*

In the feasible point algorithm, we minimize the sum of squared distances to all sets, with each set implicitly given equal $1/m$ weight. The weighted projection approach assigns a variable weight $\gamma_i \geq 0$ to each set Ω_i, (with $\sum \gamma_i = 1$.) In this case the sum of squares at x_k is minimized by the update:

$$x_{k+1} = \sum_{i=1}^m \gamma_k^i P(x_k, \Omega_i)$$

But now we wish to penalize distance only to the most distant sets, and ignore the others. So at each iteration the weights are updated: all sets shift some of their weight to a most distant set. The weights are updated as follows:

$$\gamma_{k+1}^i = \begin{cases} \gamma_k^i + \sum_i s_k^i & \text{if } \Omega_i \text{ is the most distant set from } x_k \\ \gamma_k^i - s_k^i, & \text{otherwise} \end{cases}$$

$$s_k^i = \min\{c, \gamma_k^i\}$$

We may start with $c = 1/m$, and decrease it for a more precise search. Each update moves towards the last set to be most distant. This causes the iterates to bounce back and forth between the most distant sets, but in a finite region. As c approaches zero, this region constricts to a point.

Results

We compare the 4 methods in various dimensions for various numbers of balls, with centers and radii chosen by a pseudo-random number generator. Each is given 5000 iterations.

sets; dimensions	50;5	5;50	500;500	100;1000
Subgradient	74.142	175.88	663.94	922.146
Log Sum Exp	72.910	171.87	656.04	910.453
Expanding Sets	72.994	171.87	659.50	910.828
Weights	72.910	171.87	656.10	910.457

Table 1: Max distance function values after 5000 iterations.

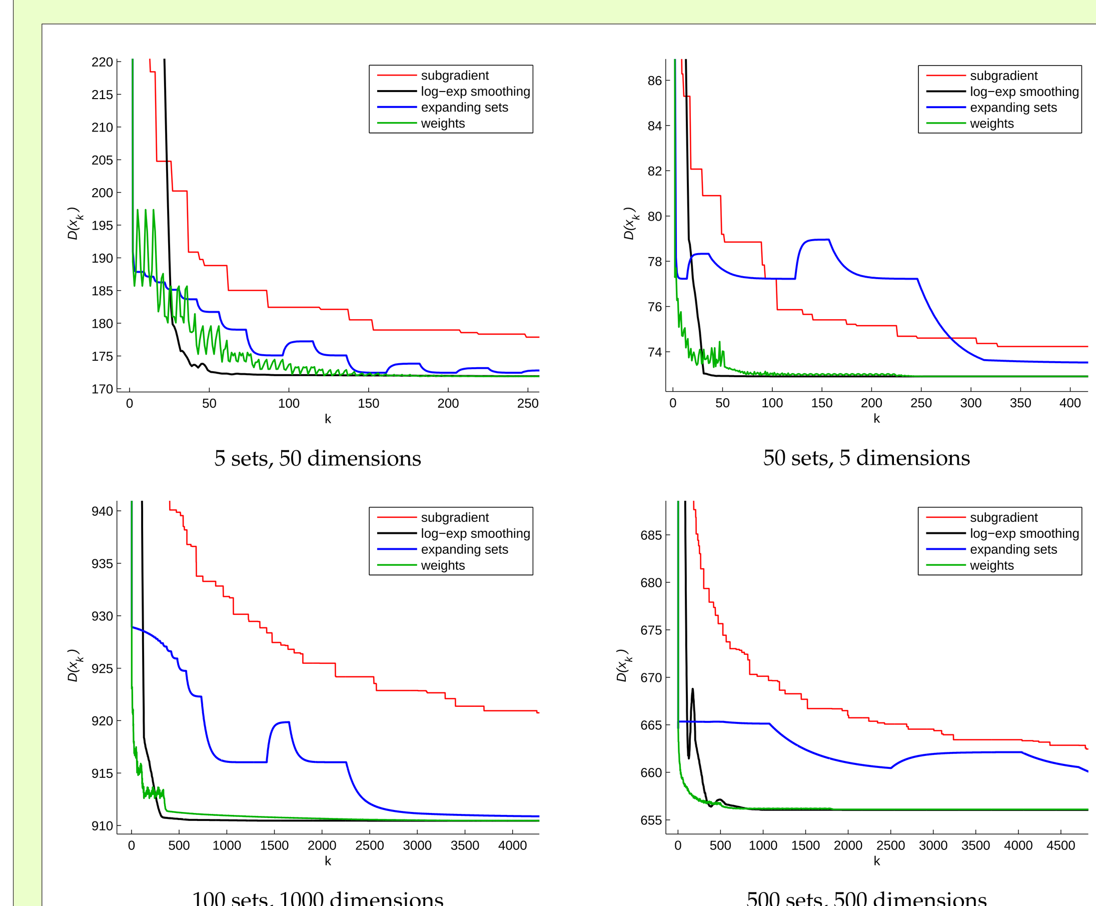


Figure 6: Function value vs iteration number for various scenarios. The subgradient (whose plot tracks best value yet attained) is consistently outperformed. The descent regions of the set expansion plot correspond to x_k being overtaken by a set. The weighted projection performs competitively with the log-smoothing method, especially with larger number of sets and dimensions.

Discussion

The four algorithms all find reasonable approximations to the optimal solution, but is one superlative? To answer this requires an analysis of convergence rates, which give the error as a function of iteration number. In the meantime it is difficult to make explicit comparisons between them. The tests on randomly generated sets offer a qualitative sense of performance, but we cannot assume that the algorithms will perform similarly for any collection of sets. Further, each implementation involves parameters which can affect their performance. With log-exponential smoothing, for example, how accurately should we minimize each iteration of the surrogate before updating? The set expansion approach relies on a locating a point in an intersection— are there more efficient algorithms for doing so? The weighted projection method appears promising, but can its convergence be proven? In any case, each of the distinct approaches may prove valuable in solving other optimization problems.

References

N.T. An, D. Giles, N.M Nam, R.B. Rector. *Log-exponential smoothing techniques and Nesterov's accelerated gradient method for generalized Sylvester problems*. arXiv:1303.7247v5 (2015)

S. Boyd. *Subgradient Methods*. Notes for EE364b, Stanford University (2014)

E. C. Chi, H. Zhou, K. Lange. *Distance majorization and its applications*. Mathematical Programming Ser. A **146**, 409-436 (2014)

Y. Nesterov. *Smooth minimization of non-smooth functions*. Mathematical Programming Ser. A **103**, 127-152 (2005)

This research was partially supported by the USA National Science Foundation under grant DMS-1411817.