EPiC Series in Computing
Volume 90, 2022, Pages 1–17
Proceedings of 9th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH22)

Implementation of Ellipsoidal Operations in CORA 2022

Victor Gaßmann and Matthias Althoff

Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany
{victor.gassmann,althoff}@in.tum.de

Abstract

Tool presentation: Ellipsoids are a broadly applied set representation for formal analyses, such as set-based observers, or reachability analysis. While ellipsoids are not closed under frequently used set operations like Minkowski sum or intersection, their widespread popularity can be attributed to their compact numerical representation and intuitive nature. As a result, there already exist toolboxes that implement many operations for ellipsoids. That said, most are either no longer maintained, or implement only a subset of necessary ellipsoidal operations. Thus, we implement all common set operations, as well as recent research results on ellipsoids as a class in the Continuous Reachability Analyzer (CORA) - a free MATLAB toolbox for continuous reachability analysis. Previously, CORA already contained implementations for many ellipsoidal operations, which, however, were lacking in speed, accuracy, and functionality. Here, we describe the implementation of the ellipsoid class in CORA and compare it against the popular, but no longer maintained, Ellipsoidal Toolbox (ET).

1 Introduction

While ellipsoids are by design constrained to centrally symmetric, convex sets, they are widely used for formal verification tasks, such as set-based observers [8], invariant region computation [1, 11], and reachability analysis [3, 9]. This can be in part attributed to their simple representation and intuitive nature.

As a result, there exist toolboxes which implement the ellipsoidal operations necessary for these aforementioned applications, among which the Ellipsoidal Toolbox (ET) [10] is arguably the most popular. The ET implements ellipsoidal operations as well as reachability analysis using ellipsoids. However, while it includes a wide variety of ellipsoidal functions, it is no longer maintained. Furthermore, many applications often additionally require the usage of other set representations to avoid overly conservative results, e.g., when operations are required which cannot be exactly computed for ellipsoids. Therefore, we include the ellipsoidal functionality from the ET into the Continuous Reachability Analyzer (CORA) [2], a MATLAB toolbox for reachability analysis, which already implements many different set representations.

In Sec. 2, we first define ellipsoids. We then describe the implementation of the ellipsoidal operations in Continuous Reachability Analyzer (CORA) in Sec. 3, and finally present a comparison of the CORA implementation with the most recent implementation of the ET in Sec. 4.

2 Preliminaries

Generally, we denote sets by calligraphic letters ($\mathcal{X}$), vectors and scalars are denoted by lowercase letters (x), and matrices are denoted by capital letters (X). Further, $\mathbb{S}$ is the set of all symmetric matrices, and $\mathbb{S}_+$ denotes the cone of all symmetric, positive semi-definite matrices. For a given vector $x \in \mathbb{R}^n$, its i-th component is denoted by x_i. We denote with $\square_{n\times m}$ an $(n \times m)$-dimensional block of $\square$, and introduce $\square_n$ as a shorthand for $\square_{n\times n}$. Further, the supremum of a function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is given by $\sup_x f(x)$. Lastly, for $x \in \mathbb{R}$, $\operatorname{ceil}(x)$ returns the smallest integer $z \in \mathbb{Z}$ for which $x \leq z$. Before defining an ellipsoid, let us first introduce support functions.

Definition 1 (Support Function [10, Sec. 2])**.** *The support function for a set $\mathcal{X} \subseteq \mathbb{R}^n$ and a direction $l \in \mathbb{R}^n$ is*

$$\rho_{\mathcal{X}}(l) = \sup_{x\in\mathcal{X}} l^T x.$$

An ellipsoid can be defined using a support function as follows.

Definition 2 (Ellipsoid [10, Sec. 2])**.** *An ellipsoid $\mathcal{E}(Q, q)$ with center $q \in \mathbb{R}^n$ and shape matrix $Q \in \mathbb{S}_+^{n\times n}$ can be defined through its support function*

$$\rho_{\mathcal{E}}(l) = l^T q + \sqrt{l^T Q l}.$$

When $\mathcal{E}(Q, q)$ is non-degenerate, i.e., $\operatorname{rank}(Q) = n$, one commonly uses the more intuitive definition

$$\mathcal{E}(Q, q) = \left\{ x \in \mathbb{R}^n \,\middle|\, (x - q)^T Q^{-1} (x - q) \leq 1 \right\}.$$

In this work, we define the dimension of an ellipsoid as the dimension of the vector space in which the user has specified the ellipsoid. Further, the minimum dimension of an ellipsoid refers to the dimension of the highest-dimensional vector space in which said ellipsoid has non-zero volume. An ellipsoid is then degenerate if $\operatorname{rank}(Q) < n$.

3 Implementation in CORA

Table 1 shows relevant ellipsoidal functions implemented in CORA, how to construct ellipsoid objects, and describes the properties of an ellipsoid object. Since CORA is implemented in MATLAB, we choose the function names to overload built-in MATLAB functions where appropriate. By default, overloaded variants of operations always compute an outer approximation. The two examples in the next section illustrate the usage of ellipsoids in CORA. As shown in table 1, ellipsoids in CORA are constructed from a shape matrix $Q \in \mathbb{S}_+^{n\times n}$ and (optionally) a center $q \in \mathbb{R}^n$ (zero by default).

Inner and Outer Ellipsoid-Ellipsoid Intersection The following code defines two ellipsoids, computes both the inner and outer ellipsoid approximations of their intersection, and plots the results:

```
% define both ellipsoids
E1 = ellipsoid([0.2,0.3;0.3,1.5],[-1;0.5]);
E2 = ellipsoid([3,-0.5;-0.5,1],[0.2;0.1]);

% compute inner and outer approximations for intersection
```

```
Eo = E1 & E2; % CORA computes outer approximations by default
Ei = and(E1,E2,'inner');

% plot results
figure; hold on
plot(E1,[1,2],'b'); plot(E2,[1,2],'r');
plot(Eo,[1,2],'k');
plot(Ei,[1,2],'--k');
xlabel('$x_1$','interpreter','latex');
ylabel('$x_2$','interpreter','latex');
```

The output is shown in Fig. 1a.

Ellipsoidal Enclosure of Degenerate Point Cloud Here, we first define random points on a hyperplane, compute the corresponding minimum-volume ellipsoid, and plot the result:

```
% define hyperplane normal
a = [1;1;-2];
% define center shift (a'*x = b)
b = 1;
% define hyperplane
hyp = conHyperplane(a,b);
% get random points on that hyperplane
V = randPoint(hyp,10);

% get minimum-volume enclosing ellipsoid
E = ellipsoid.enclosePoints(V,'min-vol');

% plot hyperplane in 3D
figure; hold on
A = a(1); B = a(2); C = a(3); D = -b;
lin_x = linspace(min(V(1,:)),max(V(1,:)),10);
lin_y = linspace(min(V(2,:)),max(V(2,:)),10);
[x,y] = meshgrid(lin_x,lin_y);
z = -1/C*(A*x+B*y+D);
grid on
surf(x,y,z,'FaceColor','none');

% plot points
plot3(V(1,:),V(2,:),V(3,:),'xr');

% plot ellipsoid boundary in 3D
V_E = randPoint(E,100,'extreme');
V_E = [V_E,V_E(:,1)];
plot3(V_E(1,:),V_E(2,:),V_E(3,:),'b');
xlabel('$x$','interpreter','latex');
ylabel('$y$','interpreter','latex');
zlabel('$z$','interpreter','latex');
```

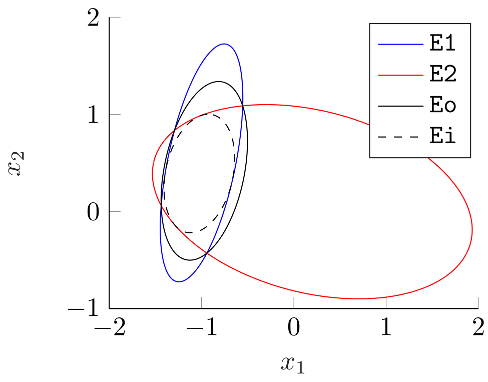
A possible output is shown in Fig. 1b.

Table 1: List of important CORA functions. Here, `E`, `P`, `H`, `h`, `v` denote an ellipsoid, a polytope, a half-space, a hyperplane, and a vector, respectively. Further, "`...`" denotes name-value pair arguments, $\{\cdot\}$ represents a list of possible values for that input argument, `'inner'` denotes an inner approximation, and `'outer'` denotes an outer approximation.

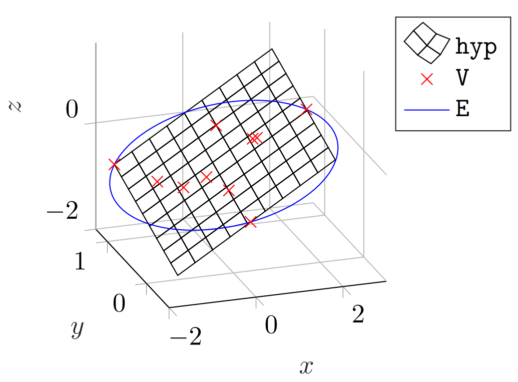
CORA function	**in ET**	**description**
Constructor		

`obj = ellipsoid(Q)`
`obj = ellipsoid(Q,q)`
	✓	Constructor: `Q`$\in \mathbb{S}_{+}^{n\times n}$, `q`$\in \mathbb{R}^n$; Properties `obj.Q` (shape matrix), `obj.q` (center)

Returns Ellipsoid		
`and(obj,{E,P,H,h,v},{'inner','outer'})`	✓	Intersection
`cartProd(obj,E)`		Outer approximation of Cartesian product
`enclosePoints(V)`	✓	Outer approximation of point cloud `V` [5]
`in(obj,{E,v})`	✓	Containment: `{E,v}` $\in$ `obj`
`ellipsoid.generateRandom(...)`	✓	Generate a random ellipsoid
`minkDiff(obj,E,L,{'inner','outer'})`	✓	Minkowski difference between two ellipsoids using matrix of directions `L`
`mtimes(obj,A)`	✓	Linear Transformation of `obj` by matrix `A`
`or(obj,{E,v},{'inner','outer'})`	✓	Union
`plus(obj,{E,v},L,{'inner','outer'})`	✓	Minkowski sum between two ellipsoids using matrix of directions `L` (also see Sec. 4.1.4)
`plus(obj,{E,v},{'outer','outer:halder'})`		Minkowski sum (also see Sec. 4.1.4; [7] for `'outer:halder'`)
Returns Value		
`distance(obj,{E,P,H,h,v})`	(✓)	Ellipsoid distance (see Sec. 4.1.2)
`isBigger(obj,E)`	✓	Containment: $\mathcal{E}$(`E.Q`) $\subseteq$ $\mathcal{E}$(`obj.Q`)
`isIntersecting(obj,{E,P,H,h,v})`	✓	Check if intersecting
`norm(obj)`		Maximum Euclidean norm (see [6])
`volume(obj)`	✓	Volume
`supportFunc(obj,l)`	✓	Support function in direction `l`
Set Conversions		
`interval(obj)`		Outer interval approximation
`zonotope(obj,{'inner','outer'})`		Ellipsoid-to-zonotope conversion [6]



(a) Example of inner and outer ellipsoid approximations of an ellipsoid-ellipsoid intersection.



(b) Example of the minimum-volume ellipsoid enclosure of a degenerate point cloud.

Figure 1: Output of example scripts

4 Comparison with the Ellipsoidal Toolbox

In this section, we compare the CORA and ET implementation. CORA implements a superset of the ellipsoidal operations in the ET. The only exception hereby are implementations that are more efficient for specific dimensions (typically dimension 2 or 3). While this certainly means that these cases will not be as performant using the n-dimensional algorithms, low-dimensional ellipsoidal operations rarely are the bottleneck. Furthermore, if one is really interested in improving runtimes, implementing these routines in a more real-time-oriented language might be more appropriate.

All computations are carried out on an Intel(R) Core(TM) i7-8650U processor with 16 GB of RAM. Furthermore, both CORA and the ET use MOSEK[1] to solve all semi-definite programming (SDP) problems. All shown results are averages over $N = 20$ runs unless specified otherwise.

Next, we first shortly introduce notable implementation differences between CORA and the ET in Sec. 4.1, followed by benchmarking the most important functions in Sec. 4.2.

4.1 Notable Changes

While CORA mostly follows the implementation of the ET, some functions and functionality were changed significantly to improve robustness, runtime, and consistency.

4.1.1 Direct Modeling of Optimization Problems

While optimization modeling frameworks like YALMIP [12] or CVX[2] are essential to quickly and easily model even complex optimization problems, they mostly require re-modeling for each problem instance, i.e., transforming the specified optimization problem into a problem description that the selected solver accepts. While, e.g., YALMIP allows one to pre-transform certain models, this initial transformation to a form required by the solver still has to be executed, which – especially for low-dimensional problems – often consumes most of the runtime.

[1]https://www.mosek.com/
[2]http://cvxr.com/cvx

Thus, CORA models all optimization problems directly, i.e., it directly produces the inputs necessary for the selected solver (see appendix A for details). Whenever possible, CORA uses built-in MATLAB functions from the optimization toolbox. While commercially available solvers might offer faster runtimes, we follow the design philosophy of avoiding external (non-MATLAB) dependencies. That said, some operations on ellipsoids require the solution of a semi-definite programming (SDP) problem, which is currently not supported by MATLAB. Here, CORA supports both the commercially available solver MOSEK as well as SDPT3 [14], which is freely available.

4.1.2 Distance

Let $\mathcal{E}(Q, q)$ be a given non-degenerate ellipsoid with $Q \in \mathbb{S}_{+}^{n \times n}$, $q \in \mathbb{R}^n$, and $\mathcal{S}$ another ellipsoid of same dimension n. The ET defines the distance d of $\mathcal{E}(Q, q)$ to $\mathcal{S}$ as

$$d = \min_{s \in \mathcal{S}, x \in \mathcal{E}} \|x - s\|_2^2 . \tag{1}$$

This formulation, however, has the downside that all $x \in \mathcal{E} \cap \mathcal{S}$ are mapped to $d = 0$. Thus, when numerically evaluating (1), deciding whether $\mathcal{S}$ and $\mathcal{E}$ truly intersect is challenging for all $x \in \mathcal{E} \cap \mathcal{S}$, irrespective of the size of this intersection, since $d = 0$ is rarely achieved in practice due to floating point errors and solver tolerances.

To circumvent this, we introduce the ellipsoid norm $\|x\|_{\mathcal{E}} = \left\|Q^{-\frac{1}{2}} x\right\|_2$ and rather define d as

$$d = \min_{x \in \mathcal{S}} \|x - q\|_{\mathcal{E}}^2 - 1. \tag{2}$$

Now, $\mathcal{E}$ and $\mathcal{S}$ still intersect whenever $d \leq 0$. However, the objective function in (2) additionally gives insight about the "depth" of the intersection. Specifically, it only maps $x \in \partial(\mathcal{E} \cap \mathcal{S})$ to 0 (which may be numerically challenging), but all $x \in \text{interior}(\mathcal{E} \cap \mathcal{S})$ are mapped to progressively more negative values the closer (in the ellipsoid norm sense) they are to the ellipsoid center. Thus, deciding whether $\mathcal{E}$ and $\mathcal{S}$ intersect becomes numerically increasingly more stable as points in $\mathcal{S}$ get closer to q (again in the ellipsoid norm sense), since then d gets increasingly closer to 1.

In summary, we note that while both (1) and (2) are second-order cone programming problems and thus convex, the resulting values for d are still subject to numerical errors due to (primal and dual) feasibility tolerances and duality gap tolerances, as well as floating point inaccuracies. That said, (2) is numerically more stable due to only mapping boundary points of the intersection to $d = 0$, as previously described.

4.1.3 Handling Degenerate Sets

The ET handles degenerate ellipsoids by bloating the corresponding dimensions so that they become non-degenerate. While this bloating approach is easy to implement, it ignores that – for some operations – one can project a degenerate ellipsoid into a lower-dimensional subspace whose dimension is equal to its minimum dimension. Thus, we shortly describe subspace projections for set conversions. Finally, we sketch how the approach for single sets can be extended to intersection and containment operations.

Set Conversions Let $\mathcal{S} \subseteq \mathbb{R}^n$ be a degenerate set with minimum dimension n_t. Here, we make use of a transformation matrix $U \in \mathbb{R}^{n\times n}$ that exposes its $n - n_t$ degenerate dimensions through the linear transformation $U^T\mathcal{S}$. The transformation matrix U for an ellipsoid $\mathcal{S} = \mathcal{E}(Q, q)$ is given by the singular value decomposition

$$Q = U \begin{bmatrix} D, & 0 \\ 0, & 0 \end{bmatrix} U^T,$$

where $U \in \mathbb{R}^{n\times n}$ is an orthogonal matrix, and $D \in \mathbb{R}^{n_t\times n_t}$ is a diagonal matrix with strictly positive entries. Similarly, for a point cloud of N points $p^{(k)} \in \mathbb{R}^n$, $k \in \{1, ..., N\}$, collected in $M \in \mathbb{R}^{n\times N}$, U is again given by the singular value decomposition

$$M = U \begin{bmatrix} D, & 0 \\ 0, & 0_{n-n_t} \end{bmatrix} V^T,$$

if $M \geq n$, or

$$M = U \begin{bmatrix} D \\ 0 \end{bmatrix} V^T,$$

if $M < n$, where U, V, and D are defined as before. The set conversion can then be computed on the projected, n_t-dimensional set, and transformed back to the original space by reintroducing the values of the degenerate dimensions, and applying the inverse transformation matrix U. In CORA, this is, e.g., applicable to functions `enclosePoints`, `interval`, or `zonotope` (see table 1 for details on arguments).

Intersection and Containment For both intersection (`and` in CORA) and containment (`in` in CORA) operations, we additionally have a non-degenerate ellipsoid $\mathcal{W} \subseteq \mathbb{R}^n$. By applying the same aformentioned transformation to both sets, i.e., $U^T\mathcal{S}$ and $U^T\mathcal{W}$, it is clear that $U^T\mathcal{S} \cap U^T\mathcal{W} = U^T(\mathcal{S} \cap \mathcal{W})$ so that we can equally use the same transformations as shown above. Obviously, for the containment check, no back-transformation is required.

4.1.4 Plus

The ET implements the Minkowski sum only for user-defined directions. While one can compute a union or intersection (inner or outer approximations) of the resulting ellipsoids of multiple directions, this is numerically expensive. Thus, for outer approximations, CORA additionally implements an SDP problem computing a minimum-volume ellipsoid covering the Minkowski sum of ellipsoids [4]. Furthermore, we also implement the approximate but performant approach from [7], which computes an optimal, outer approximation based on a chosen parameterization of the resulting ellipsoid.

4.2 Results

To compare volumes (where appropriate) of an ellipsoid $\mathcal{E}(Q, q)$, and to compare runtimes, we introduce

$$V_{\text{rel}} = \left(\frac{\det(Q_{\text{ET}})}{\det(Q_{\text{CORA}})}\right)^{\frac{1}{n}},$$
$$T_{\text{rel}} = \frac{T_{\text{ET}}}{T_{\text{CORA}}},$$

where $\det\left(Q_{(\cdot)}\right)^{\frac{1}{n}}$ is a measure proportional to the normalized volume of the corresponding ellipsoid. This normalization is achieved by the $\frac{1}{n}$ exponent, which avoids exponential increase/decrease of $\det\left(Q_{(\cdot)}\right)$ with increasing dimension for large/small ellipsoids.

Table 2 compares the implementation of various functions in CORA and the ET. For each benchmarked function, results for non-degenerate sets (above dashed line) and for degenerate sets are shown (below dashed line). The minimum dimension of all degenerate sets is $n - \mathrm{ceil}\,(n/2)$ (where n is the dimension of the set). Whenever no numerical issues occur for a given toolbox implementation, or if results are identical ($V_{\mathrm{rel}} = 1$), we omitted the corresponding row in table 2. The only exception here is the relative volume ratio for degenerate cases, as V_{rel} is undefined for degenerate sets.

We conclude that CORA is often much faster for smaller dimensions, and mildly faster for higher dimensions when sets are non-degenerate. An exception hereby is the `and(obj,E,'outer')` operation. For non-degenerate cases and operations which require the solution of a semi-definite programming (SDP) problem, these speed-ups can be mostly attributed to the direct modeling of optimization problems (see Sec. 4.1.1). Improvements for cases where no SDP problems have to be solved are mostly dominated by an overall more efficient architecture of CORA (see, e.g., `plus(obj,p)` in table 2, which essentially only requires the addition of two vectors). Further, for degenerate cases, we observe that runtime improvements start to dominate overall runtime only for larger-dimensional sets. That is expected since the transformation to the lower-dimensional subspaces (see Sec. 4.1.3) initially takes some time, but saves a lot of computational effort for high-dimensional problems, which are computationally expensive (such as SDP problems). For the same reason, handling of degenerate cases can even negatively affect runtime when the operation itself is computationally cheap (see, e.g., `and(obj,E,'outer')`). That said, while volume ratios for the degenerate cases are not shown due to the 0 volume of degenerate sets, the explicit handling in CORA still produces more accurate results by design, as we do not bloat degenerate dimensions. Lastly, we notice that for the degenerate cases of `in(obj,E)` in table 2, the ET has a failure rate of almost 50 %. For each run of this benchmark, we alternate between generating a contained pair of ellipsoids and a non-intersecting pair of ellipsoids; the ET fails to identify this degenerate containment almost every time.

Table 2: Comparison of the CORA and ET implementation of various functions. For each function, we show the averages for non-degenerate cases (above dashed line) and averages for degenerate cases (below dashed line). "Num. issues rate" hereby denotes the number of failed runs (due to numerical issues) over the total number of runs (per dimension).

Dimensions	**2**	**5**	**10**	**20**	**30**	**40**
`and(obj,E,'inner')`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	59.46	36.37	19.07	4.55	2.00	1.30
Num. issues rate (ET)	0	0	0	0	0.05	0
T_{rel}	72.93	24.39	14.91	9.78	10.53	13.67
`and(obj,E,'outer')`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	1.13	1.20	1.14	0.87	0.80	0.60
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	1.07	0.62	0.42	0.29	0.23	0.15
`enclosePoints(V)`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	111.55	91.11	46.33	14.23	5.35	2.43
ET could not handle degeneracies						
`in(obj,E)`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	100.77	100.22	65.52	65.61	45.48	29.27
Num. issues rate (ET)	0.50	0.50	0.50	0.45	0.50	0.50
T_{rel}	114.97	74.99	56.01	65.84	50.32	45.31
`isIntersecting(obj,E)`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	1.14	4.69	9.23	19.11	24.61	33.21
Num. issues rate (ET)	0	0	0.20	0.20	0.35	0.20
T_{rel}	25.88	31.61	23.14	26.66	15.97	34.83
`lminkDiff(obj,E,L,'inner')`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	7.10	10.90	6.08	4.26	2.90	2.47
Not implemented for CORA or ET						
`lminkDiff(obj,E,L,'outer')`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	7.10	10.90	6.08	4.26	2.90	2.47
Not implemented for CORA or ET						

`or(obj,E,'outer')`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	50.68	33.54	17.78	5.72	5.72	1.33
Num. issues rate (ET)	0.30	0.50	0.60	1	1	1
T_{rel}	48.44	33.24	19.41	NaN	NaN	NaN
`plus(obj,v)`						
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	3.95	3.62	4.29	5.73	5.37	3.60
Num. issues rate (ET)	0	0	0	0	0	0
T_{rel}	3.68	3.49	4.45	5.13	5.53	3.46

5 Conclusions

In this paper, we present the implementation of ellipsoidal operations in the Continuous Reachability Analyzer (CORA) and compare it to the popular Ellipsoidal Toolbox (ET). CORA not only implements virtually all ellipsoidal operations offered by the ET, but also incorporates both well-known results not implemented in the ET, as well as more recent results from the literature. By modeling optimization problems directly without using a modeling framework, we achieve a significant speed-up compared to the ET for almost all operations, especially for low-dimensional problems. Furthermore, ellipsoidal operations in CORA now explicitly handle degenerate sets, which – as we have shown – increases accuracy of affected operations and reduces computation times significantly.

6 Acknowledgments

We gratefully acknowledge financial support by the project justITSELF funded by the European Research Council (ERC) under grant agreement No 817629.

A Dual Problem for SDP Modeling

Since many ellipsoidal operations require the solution to an SDP problem, we need to bring a given SDP problem to a form that the given solver accepts. Therefore, in appendix A.1 we first briefly introduce all SDP problems in CORA and their corresponding ellipsoidal operation. To see that the dual problem is very useful for matching the introduced SDP problems to a standard problem formulation of a solver, we will derive the dual problem of MOSEK's primal standard problem in appendix A.2. While this is not a novel contribution, to the best knowledge of the authors this information is not compactly available, and thus we include it here to ease understanding of the SDP problems implemented in CORA. We use the MOSEK primal problem formulation rather than the SDPT3 formulation simply because we focused on MOSEK in this paper. Lastly, we derive a special case of the dual problem in appendix A.3, and also demonstrate – using the example of the ellipsoid-ellipsoid containment problem – that identifying the necessary problem matrices required by the solver is possible using this special case of the dual problem.

A.1 SDP Problems in CORA

Here, we introduce all SDP problems along with their corresponding ellipsoidal operation. For convenience, we define M non-degenerate ellipsoids

$$\begin{aligned}\mathcal{E}\left(Q^{(i)}, q^{(i)}\right) &= \left\{x \in \mathbb{R}^n \,\middle|\, \left(x - q^{(i)}\right)^T {Q^{(i)}}^{-1} \left(x - q^{(i)}\right) \leq 1\right\} \\ &= \left\{x \in \mathbb{R}^n \,\middle|\, x^T \underbrace{{Q^{(i)}}^{-1}}_{:=A^{(i)}} x + 2 \underbrace{\left(-{q^{(i)}}^T {Q^{(i)}}^{-1}\right)}_{:={b^{(i)}}^T} x + \underbrace{{q^{(i)}}^T {Q^{(i)}}^{-1} q^{(i)} - 1}_{c^{(i)}} \leq 0\right\}, \end{aligned} \quad (3)$$

with $Q^{(i)} \in \mathbb{S}_+^{n\times n}$ and $q^{(i)} \in \mathbb{R}^n$, where $1 \leq i \leq M$, and introduce I as the identity matrix.

A.1.1 Inner Approximation of Intersection

Given are M non-degenerate ellipsoids as in (3). The inner approximation of the intersection of all ellipsoids $\mathcal{E}\left(Q^{(i)}, q^{(i)}\right)$, $1 \leq i \leq M$, is given by $\mathcal{E}\left(\hat{B}^2, \hat{q}\right)$, where [5, Sec. 8.4.2]

$$\begin{aligned}\hat{B}, \hat{q} = \arg &\min_{B,q,\lambda} \log\det B^{-1}, \\ &\text{s.t.} \begin{bmatrix} -\lambda_i - c_i + {b^{(i)}}^T {A^{(i)}}^{-1} b^{(i)}, & 0, & \left(q + {A^{(i)}}^{-1} b^{(i)}\right)^T \\ 0, & \lambda_i I_n, & B \\ q + {A^{(i)}}^{-1} b^{(i)}, & B, & {A^{(i)}}^{-1} \end{bmatrix} \succeq 0,\ \forall i \in \{1, ..., M\}, \\ &B \succeq 0.\end{aligned}$$

A.1.2 Outer Approximation of Minkowski Sum

Given are M non-degenerate ellipsoids as in (3). The outer approximation of the Minkowski sum of all ellipsoids $\mathcal{E}\left(Q^{(i)}, q^{(i)}\right)$, $1 \leq i \leq M$, is given by $\mathcal{E}\left(\hat{W}^{-1}, -\hat{W}^{-1}\hat{b}\right)$, with [4, Sec. 3.7.4]

$$\begin{aligned}\hat{W}, \hat{b} = \arg &\min_{W,b,\tau} \log\det W^{-1}, \\ &\text{s.t.} \begin{bmatrix} {E^{(0)}}^T W E^{(0)}, & {E^{(0)}}^T b, & 0 \\ \left({E^{(0)}}^T b\right)^T, & -1, & b^T \\ 0, & b, & -W \end{bmatrix} - \sum_{i=1}^{M} \tau_i \begin{bmatrix} \tilde{A}^{(i)}, & \tilde{b}^{(i)}, & 0 \\ {\tilde{b}^{(i)}}^T, & c^{(i)}, & 0 \\ 0, & 0, & 0 \end{bmatrix} \preceq 0, \\ &\tau_i \geq 0,\ \forall i \in \{1, ..., M\}, \\ &W \succeq 0,\end{aligned}$$

where

$$\begin{aligned} E^{(i)} &= \begin{bmatrix} 0_{n\times((i-1)n)}, & I_{n\times n}, & 0_{n\times((N-i)n)} \end{bmatrix},\ 1 \leq i \leq M, \\ E^{(0)} &= \sum_{i=1}^{M} E^{(i)}, \\ \tilde{A}^{(i)} &= {E^{(i)}}^T A^{(i)} E^{(i)}, \\ \tilde{b}^{(i)} &= {E^{(i)}}^T b^{(i)}. \end{aligned}$$

A.1.3 Outer Approximation of Union

Given are M non-degenerate ellipsoids as in (3). The outer approximation of the union of all ellipsoids $\mathcal{E}\left(Q^{(i)}, q^{(i)}\right)$, $1 \leq i \leq M$, is given by $\mathcal{E}\left(\hat{A}^{-2}, -\hat{A}^{-2}\hat{\tilde{b}}\right)$, where [5, Sec. 8.4.1]

$$\begin{aligned}
\hat{A}, \hat{\tilde{b}} = \arg \min_{A, \tilde{b}, \tau} & \log\det A^{-1}, \\
\text{s.t.} & \begin{bmatrix} A^2 - \tau_i A^{(i)}, & \tilde{b} - \tau_i b^{(i)}, & 0 \\ \left(\tilde{b} - \tau_i b^{(i)}\right)^T, & -1 - \tau_i c^{(i)}, & \tilde{b}^T \\ 0, & \tilde{b}, & -A^2 \end{bmatrix} \preceq 0, \ \forall i \in \{1, ..., M\}, \\
& \tau_i \geq 0, \ \forall i \in \{1, ..., M\}, \\
& A \succeq 0,
\end{aligned}$$

which is convex in $A^2 \in \mathbb{S}_+^{n\times n}$ $\tilde{b} \in \mathbb{R}^n$, and $\tau \in \mathbb{R}^M$.

A.1.4 Outer Approximation of Point Enclosure

Given are points $x^{(i)} \in \mathbb{R}^n$, $i \in \{1, ..., M\}$. Then the minimum-volume ellipsoid covering all points $x^{(i)}$ is given by $\mathcal{E}\left(\hat{E}^{-2}, -\hat{E}^{-1}\hat{d}\right)$, where [5, Sec. 8.4.1]

$$\begin{aligned}
\hat{E}, \hat{d} = \arg \min_{E, d} & \log\det E^{-1}, \\
\text{s.t.} & \left\lVert E x^{(i)} + d \right\rVert_2 \leq 1, \ \forall i \in \{1, ..., M\}, \\
& E \succeq 0.
\end{aligned}$$

A.1.5 Inscribed Polytope

For a given polytope $\mathcal{P} = \left\{x \in \mathbb{R}^n \,\middle|\, {a^{(i)}}^T x \leq w_i, \ 1 \leq i \leq M\right\}$, $a^{(i)} \in \mathbb{R}^n$, $w_i \in \mathbb{R}$, the maximum-volume inscribed ellipsoid $\mathcal{E}\left(\hat{B}^2, \hat{q}\right)$ is given by [5, Sec. 8.4.2]

$$\begin{aligned}
\hat{B}, \hat{q} = \arg \min_{B, q} & \log\det B^{-1}, \\
\text{s.t.} & \left\lVert B a^{(i)} \right\rVert_2 + {a^{(i)}}^T q \leq w_i, \ \forall i \in \{1, ..., M\}, \\
& B \succeq 0.
\end{aligned}$$

A.1.6 Ellipsoid-Ellipsoid Containment

Given two non-degenerate ellipsoids $\mathcal{E}(Q, q)$, $Q \in \mathbb{S}_+^{n\times n}$, $q \in \mathbb{R}^n$, and $\mathcal{E}(W, c)$, $W \in \mathbb{S}_+^{n\times n}$, $c \in \mathbb{R}^n$, it holds that $\mathcal{E}(W, c) \subseteq \mathcal{E}(Q, q)$ if and only if there exists a $t \in \mathbb{R}$ such that

$$-\begin{bmatrix} Q^{-1}, & -Q^{-1}q \\ -\left(Q^{-1}q\right)^T, & q^T Q^{-1} q - 1 \end{bmatrix} + t \begin{bmatrix} W^{-1}, & -W^{-1}c \\ -\left(W^{-1}c\right)^T, & c^T W^{-1} c - 1 \end{bmatrix} \succeq 0, \tag{4a}$$

$$t > 0. \tag{4b}$$

Since (4) contains a strict inequality, it does not lend itself very well to numerical evaluation. Instead, for a given user-specified tolerance $\delta > 0$, we introduce

$$-\begin{bmatrix} Q^{-1}, & -Q^{-1}q \\ -\left(Q^{-1}q\right)^T, & q^TQ^{-1}q-1 \end{bmatrix} + t\begin{bmatrix} W^{-1}, & -W^{-1}c \\ -\left(W^{-1}c\right)^T, & c^TW^{-1}c-1 \end{bmatrix} \succeq 0, \tag{5a}$$

$$t \geq \delta. \tag{5b}$$

A.2 Primal to Dual

MOSEK can solve problems of the form[3]

$$p^* = \min_{x,X^{(i)}} \quad c^Tx + \sum_{i=1}^{N} \left\langle C^{(i)}, X^{(i)} \right\rangle, \tag{6a}$$

$$\text{s.t.} \quad \underline{b}_j \leq {a^{(j)}}^T x + \sum_{i=1}^{N} \left\langle A^{(ij)}, X^{(i)} \right\rangle \leq \bar{b}_j,\ j \in \{1, ..., M\}, \tag{6b}$$

$$\underline{d} \leq x \leq \bar{d}, \tag{6c}$$

$$x \in \mathcal{K}, \tag{6d}$$

$$X^{(i)} \succeq 0,\ i \in \{1, ..., N\}, \tag{6e}$$

where $p^* \in \mathbb{R}$ is the optimal primal objective value, $x \in \mathbb{R}^n$ contains n scalar optimization variables, $X^{(i)} \in \mathbb{S}_+^{n\times n}$ are N positive semi-definite optimization matrices, $c \in \mathbb{R}^n$, $C^{(i)} \in \mathbb{S}^{n\times n}$, $\left\langle C^{(i)}, X^{(i)} \right\rangle = \sum_{k,l} C_{kl}^{(i)} X_{kl}^{(i)}$ denotes the Frobenius inner product ($\square_{kl}$ denotes the element in the k-th row and l-th column), $\underline{b} \in \mathbb{R}^M$, $\bar{b} \in \mathbb{R}^M$, $a^{(j)} \in \mathbb{R}^n$, $A^T = \begin{bmatrix} a^{(1)}, & \dots, & a^{(M)} \end{bmatrix} \in \mathbb{R}^{n\times M}$, $A^{(ij)} \in \mathbb{S}^{n\times n}$, $\underline{d} \in \mathbb{R}^n$, $\bar{d} \in \mathbb{R}^n$, and $\mathcal{K} \subseteq \mathbb{R}^n$ is the Cartesian product of convex cones supported by MOSEK.

When comparing the structure of (6) to the SDP problems introduced in appendix A.1, we notice that they are structurally quite different, since, e.g., (6) does not offer Linear Matrix Inequality (LMI) constraints, which are, e.g., required for the containment problem in (4). As we will see, the dual of (6) will be closer in structure to SDP problems introduced in appendix A.1, and thus we derive the dual problem subsequently.

[3]MOSEK MATLAB documentation (Sec. 6.6): `https://docs.mosek.com/9.3/toolbox.pdf`

To that end, we first construct the dual function of (6), which is given by

$$\Theta\left(\underline{\mu},\bar{\mu},\underline{\eta},\bar{\eta},s,S^{(i)}\right) =$$

$$\inf_{x,X^{(i)}} \left\{ c^T x + \sum_{i=1}^{N} \left\langle C^{(i)}, X^{(i)} \right\rangle \right. \tag{7a}$$

$$+ \sum_{j=1}^{M} \underline{\mu}_j \left(\underline{b}_j - {a^{(j)}}^T x - \sum_{i=1}^{N} \left\langle A^{(ij)}, X^{(i)} \right\rangle \right) + \sum_{j=1}^{M} \bar{\mu}_j \left(-\bar{b}_j + {a^{(j)}}^T x + \sum_{i=1}^{N} \left\langle A^{(ij)}, X^{(i)} \right\rangle \right) \tag{7b}$$

$$+ \underline{\eta}^T \left(\underline{d} - x\right) + \bar{\eta}^T \left(x - \bar{d}\right) \tag{7c}$$

$$- s^T x \tag{7d}$$

$$\left. - \sum_{i=1}^{N} \left\langle S^{(i)}, X^{(i)} \right\rangle \right\}, \tag{7e}$$

where each line in (7) aligns with objective function and constraints in (6), and where $\underline{\mu} \in \mathbb{R}^M$, $\bar{\mu} \in \mathbb{R}^M$, $\underline{\eta} \in \mathbb{R}^n$, $\bar{\eta} \in \mathbb{R}^n$, $s \in \mathbb{R}^n$, and $S^{(i)} \in \mathbb{R}_+^{n\times n}$, $i \in \{1,...,N\}$. While most of the dual function follows from standard duality theory, we briefly discuss (7d) and (7e). For any feasible x and $X^{(i)}$ of (6), $\underline{\mu} \geq 0$, $\bar{\mu} \geq 0$, $\underline{\eta} \geq 0$, $\bar{\eta} \geq 0$, $s \in \operatorname{dual}(\mathcal{K})$, and $S^{(i)} \in \mathbb{S}_+^{n\times n}$, weak duality holds, i.e., $\Theta\left(\underline{\mu},\bar{\mu},\underline{\eta},\bar{\eta},s,S^{(i)}\right) \leq p^*$: The dual cone of $\mathcal{K}$ is given by $\operatorname{dual}(\mathcal{K}) = \left\{z \in \mathbb{R}^n \mid z^T x \geq 0,\ x \in \mathcal{K}\right\}$. Thus, since $s^T x \geq 0$ for all feasible x and $\forall s \in \operatorname{dual}(\mathcal{K})$, we subtract $s^T x$ from (7) to ensure $\Theta\left(\underline{\mu},\bar{\mu},\underline{\eta},\bar{\eta},s,S^{(i)}\right) \leq p^*$ (weak duality)[4]. Similarly, the dual cone of the positive semi-definite cone $\mathbb{S}_+^{n\times n}$ is given by $\operatorname{dual}\left(\mathbb{S}_+^{n\times n}\right) = \left\{Z \in \mathbb{S}^{n\times n} \mid \langle Z, X\rangle \geq 0,\ X \in \mathbb{S}_+^{n\times n}\right\} = \mathbb{S}_+^{n\times n}$ (since the cone of positive semi-definite matrices is self-dual), thus $\left\langle S^{(i)}, X^{(i)}\right\rangle \geq 0$ for all feasible $X^{(i)}$ and $\forall S^{(i)} \in \mathbb{S}_+^{n\times n}$, and therefore we also subtract $\sum_{i=1}^{N} \left\langle S^{(i)}, X^{(i)}\right\rangle$ from (7).

As a next step, we collect all terms in (7) involving x and $X^{(i)}$, which yields

$$\begin{aligned} &\Theta\left(\underline{\mu},\bar{\mu},\underline{\eta},\bar{\eta},s,S^{(i)}\right) = \\ &\inf_{x,X^{(i)}} \left\{ x^T \left(c - A^T\underline{\mu} + A^T\bar{\mu} - \underline{\eta} + \bar{\eta} - s\right) \right. \\ &\left. + \sum_{i=1}^{N} \left\langle C^{(i)} - \sum_{j=1}^{M} A^{(ij)}\left(\underline{\mu}_j - \bar{\mu}_j\right) - S^{(i)}, X^{(i)} \right\rangle \right\} \\ &+ \underline{b}^T\underline{\mu} - \bar{b}^T\bar{\mu} + \underline{d}^T\underline{\eta} - \bar{d}^T\bar{\eta}. \end{aligned} \tag{8}$$

[4] MOSEK cookbox (3.3.0), Sec. 8.3: https://docs.mosek.com/MOSEKModelingCookbook-letter.pdf

Since we included all constraints on both x and $X^{(i)}$ in the Lagrangian function, (8) is unbounded below unless the factor of x^T and the left-hand side of the Frobenius inner product involving $X^{(i)}$ equal zero, i.e.

$$\Theta\left(\underline{\mu}, \bar{\mu}, \underline{\eta}, \bar{\eta}, s, S^{(i)}\right) = \tag{9}$$
$$\begin{cases} \underline{b}^T \underline{\mu} - \bar{b}^T \bar{\mu} + \underline{d}^T \underline{\eta} - \bar{d}^T \bar{\eta}, & \text{if } \begin{array}{l} c - A^T \underline{\mu} + A^T \bar{\mu} - \underline{\eta} + \bar{\eta} - s = 0 \wedge \\ C^{(i)} - \sum_{j=1}^{M} A^{(ij)} \left(\underline{\mu}_j - \bar{\mu}_j\right) = S^{(i)},\ \forall i \in \{1, ..., N\}, \end{array} \\ -\infty, & \text{otherwise.} \end{cases}$$

Hence, the dual problem is given by

$$d^* = \max_{\underline{\mu}, \bar{\mu}, \underline{\eta}, \bar{\eta}, s} \underline{b}^T \underline{\mu} - \bar{b}^T \bar{\mu} + \underline{d}^T \underline{\eta} - \bar{d}^T \bar{\eta}, \tag{10a}$$

$$\text{s.t.} \quad c - A^T \left(\underline{\mu} - \bar{\mu}\right) - \underline{\eta} + \bar{\eta} - s = 0, \tag{10b}$$

$$C^{(i)} - \sum_{j=1}^{M} A^{(ij)} \left(\underline{\mu}_j - \bar{\mu}_j\right) \succeq 0,\ i \in \{1, ..., N\}, \tag{10c}$$

$$s \in \operatorname{dual}(\mathcal{K}), \tag{10d}$$

$$\begin{bmatrix}\underline{\mu}^T, & \bar{\mu}^T, & \underline{\eta}^T, & \bar{\eta}^T\end{bmatrix} \geq 0, \tag{10e}$$

where $S^{(i)} \in \mathbb{S}_+^{n \times n}$ is implicitly considered by replacing $C^{(i)} - \sum_{j=1}^{M} A^{(ij)} \left(\underline{\mu}_j - \bar{\mu}_j\right) = S^{(i)}$ with $C^{(i)} - \sum_{j=1}^{M} A^{(ij)} \left(\underline{\mu}_j - \bar{\mu}_j\right) \succeq 0$, $i \in \{1, ..., N\}$.

Clearly, (5) is structurally more similar to (10) than (6), especially when comparing (5a) to (10c). However, we still cannot identify all necessary matrices from (10), since there is no equivalent for (5b) in (10). Therefore, we now derive a special case of the dual problem which is used for all SDP problems in CORA, and that structurally includes the containment problem.

Remark We note that contrary to linear programs, strong duality does not necessarily hold even for feasible SDP problems [13, Sec. 4] and thus technically require a sufficient constraint qualification to ensure strong duality, i.e., $d^* = p^*$. However, for practical applications of SDP, it is reasonable to assume that such a constraint qualification always holds, and that a non-zero duality gap most likely signals issues with the model itself[5].

A.3 Dual Problem with Inequality Constraints

In this section, we derive a special form of the dual problem in (10) that includes an inequality constraint, motivated by (5b). We will then use this special form of (10) to identify the necessary matrices to model the containment problem in (5).

Derivation Say we identified A, $A^{(ij)}$, c, $C^{(i)}$ from (10), $\forall i \in \{1, ..., N\}$, $j \in \{1, ..., M\}$, but additionally require the constraint $\mu_k \geq \epsilon$ (motivated by (5b)), where $\mu = \underline{\mu} - \bar{\mu}$, $1 \leq k \leq K \leq M$, and $\epsilon \in \mathbb{R}^K$. For all SDP problems in CORA, it suffices to set $b = \underline{b} = \bar{b}$, which implies $\mu = \underline{\mu} - \bar{\mu}$, i.e., (6b) becomes an equality constraint. Substituting c, A^T, $\underline{d}$, $\bar{d}$, $\underline{\eta}$, $\bar{\eta}$, s in (10) with $\hat{c} = \begin{bmatrix} c \\ -\epsilon \end{bmatrix}$, $\hat{A}^T = \begin{bmatrix} A^T & \\ -I_{K\times K}, & 0 \end{bmatrix}$, $\hat{\underline{d}} = \begin{bmatrix} \underline{d} \\ 0_K \end{bmatrix}$, $\hat{\bar{d}} = \begin{bmatrix} \bar{d} \\ \infty_K \end{bmatrix}$, $\hat{\underline{\eta}} = \begin{bmatrix} \underline{\eta} \\ \tilde{\underline{\eta}} \end{bmatrix}$, $\hat{\bar{\eta}} = \begin{bmatrix} \bar{\eta} \\ \tilde{\bar{\eta}} \end{bmatrix}$, $\hat{s} = \begin{bmatrix} s \\ \tilde{s} \end{bmatrix}$,

[5]MOSEK cookbox (3.3.0), Sec. 8.4: https://docs.mosek.com/MOSEKModelingCookbook-letter.pdf

$\hat{\mathcal{K}} = \mathcal{K} \times \tilde{\mathcal{K}}$ – where $\tilde{\underline{\eta}}$, $\tilde{\bar{\eta}}$, $\tilde{s}$ are all vectors from $\mathbb{R}^K$ and $\tilde{\mathcal{K}} \subseteq \mathbb{R}^K$ – yields

$$\max_{\mu,\underline{\eta},\bar{\eta},s} b^T\mu + \underline{d}^T\underline{\eta} - \bar{d}^T\bar{\eta}, \tag{11a}$$

$$\text{s.t.} \quad c - A^T\mu - \underline{\eta} + \bar{\eta} - s = 0, \tag{11b}$$

$$-\epsilon_k + \mu_k - \tilde{\underline{\eta}}_k - \tilde{s}_k = 0,\ k \in \{1,...,K\}, \tag{11c}$$

$$C^{(i)} - \sum_{j=1}^{M} A^{(ij)}\mu_j \succeq 0,\ i \in \{1,...,N\}, \tag{11d}$$

$$\hat{s} \in \mathrm{dual}\left(\hat{\mathcal{K}}\right), \tag{11e}$$

$$\left[\underline{\eta}^T, \quad \bar{\eta}^T, \quad \tilde{\underline{\eta}}^T\right] \geq 0, \tag{11f}$$

where $\tilde{\bar{\eta}}$ is forced to 0 due to the its coefficients being $-\infty$ in the objective function. Further, if we do not constrain the primal scalar variables $\tilde{x} \in \mathbb{R}^K$ that we introduced by this extension of c, A^T etc. with a cone in the primal problem, i.e., $\tilde{\mathcal{K}} = \mathbb{R}^K$, the corresponding dual cone is given by $\mathrm{dual}\left(\tilde{\mathcal{K}}\right) = \left\{z \in \mathbb{R}^K \,\middle|\, z^T\tilde{x} \geq 0,\ \tilde{x} \in \tilde{\mathcal{K}}\right\}$ and thus $\tilde{s} \in \mathrm{dual}\left(\tilde{\mathcal{K}}\right)$ implies $\tilde{s} = 0$. Hence (11c) becomes $\mu_k - \epsilon_k = \tilde{\underline{\eta}}_k \geq 0$. If we now further assume that $\bar{d} = -\underline{d} = \infty$, we arrive at

$$\max_{\mu,s} b^T\mu, \tag{12a}$$

$$\text{s.t.} \quad c - A^T\mu - s = 0, \tag{12b}$$

$$\mu_k \geq \epsilon_k,\ k \in \{1,...,K\}, \tag{12c}$$

$$C^{(i)} - \sum_{j=1}^{M} A^{(ij)}\mu_j \succeq 0,\ i \in \{1,...,N\}, \tag{12d}$$

$$s \in \mathrm{dual}\left(\mathcal{K}\right), \tag{12e}$$

which now structurally includes (5). We note that in addition to (5), the problem in (12) is, e.g., also relevant for modeling `in(obj,E)` (ellipsoid-in-ellipsoid containment) and `or(obj,E,'outer')` (outer approximation of union of two ellipsoids).

Identification of Problem Matrices Since (12) now includes the inequality constraint (12c), we can identify all necessary matrices by comparing (12) with (5), which yields

$$\begin{aligned}
\hat{A}^T &= -1,\\
\hat{\underline{d}} &= 0,\\
\hat{\bar{d}} &= \infty,\\
C^{(1)} &= -\begin{bmatrix} Q^{-1}, & -Q^{-1}q \\ -\left(Q^{-1}q\right)^T, & q^TQ^{-1}q - 1\end{bmatrix},\\
A^{(11)} &= -\begin{bmatrix} W^{-1}, & -W^{-1}c \\ -\left(W^{-1}c\right)^T, & c^TW^{-1}c - 1\end{bmatrix},\\
\hat{c} &= -\delta,\\
\mathcal{K} &= \mathbb{R}.
\end{aligned}$$

Since (5) is a feasibility problem, the choice of b is arbitrary.

References

[1] T. Alamo, A. Cepeda, and D. Limon. Improved computation of ellipsoidal invariant sets for saturated control systems. In *Conference on Decision and Control*, pages 6216 – 6221. IEEE, 2005.

[2] M. Althoff. An introduction to CORA 2015. In *Workshop on Applied Verification for Continuous and Hybrid Systems*, page 120–151, 2015.

[3] D. V. Balandin, R. S. Biryukov, and M. M. Kogan. Ellipsoidal reachable sets of linear time-varying continuous and discrete systems in control and estimation problems. *Automatica*, 116, 2020. article no. 108926.

[4] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[6] V. Gaßmann and M. Althoff. Scalable zonotope-ellipsoid conversions using the Euclidean zonotope norm. In *American Control Conference*, pages 4715 – 4721. IEEE, 2020.

[7] A. Halder. On the parameterized computation of minimum volume outer ellipsoid of Minkowski sum of ellipsoids. In *Conference on Decision and Control*, pages 4040 – 4045. IEEE, 2018.

[8] A. B. Kurzhanski. Hamiltonian techniques for the problem of set-membership state estimation. *International Journal of Adaptive Control and Signal Processing*, 25(3):249–263, 2010.

[9] A. B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In *Hybrid Systems: Computation and Control*, pages 202–214. Springer, 2000.

[10] A. A. Kurzhanskiy and P. Varaiya. Ellipsoidal toolbox (ET). In *Conference on Decision and Control*, pages 1498 – 1503. IEEE, 2006.

[11] Y. Li and Z. Lin. The maximal contractively invariant ellipsoids for discrete-time linear systems under saturated linear feedback. *Automatica*, 76:336–344, 2017.

[12] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *International Conference on Robotics and Automation*, pages 284 – 289. IEEE, 2004.

[13] M. J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.

[14] K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3 — A MATLAB software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1-4):545–581, 1999.