Modified Fourier-Motzkin Elimination Algorithm for Reducing Systems of Linear Inequalities with Unconstrained Parameters

Mario Bencomo
and Luis Gutierrez
University of Texas at El Paso
mjbencomo@miners.utep.edu

Martine Ceberio
University of Texas at El Paso
mceberio@utep.edu

ABSTRACT

The need for eliminating redundancies in systems of linear inequalities arises in many applications. In linear programming (LP), one seeks a solution that optimizes a given linear objective function subject to a set of linear constraints, sometimes posed as linear inequalities. Linear inequalities also arise in the context of tensor decomposition. Due to the lack of uniqueness in higher-order tensor decomposition, non-negativity constraints are imposed on the decomposition factors, yielding systems of linear inequalities. Eliminating redundancies in such systems can reduce the number of computations, and hence improve computation times in applications.

Current techniques for eliminating redundant inequalities are not viable in higher dimensions [7]. As an alternative we propose a modified version of the Fourier-Motzkin Elimination Algorithm (`ModFMEA`), implemented in MATLAB, to reduce redundancies in a given system of linear constraints over reals posed as linear inequalities, i.e., $\sum_{n=1}^{N} a_n x_n \geq b$ where a_n and b are real constants. Reduction is obtained, at each orthant containing the solution set, by taking the lower and upper bounds of x_i-normalized inequalities $x_i \geq l$ and $u \geq x_i$ respectively, where l and u are linear terms with no occurrence of x_i, for $i = 1, 2, ..., N$. The reduced system over the whole solution set can be obtained by taking the union of the reduced system at each orthant.

This method eliminates redundancies by retaining a system of linear inequalities that define the set of feasible solutions. It works under the assumption that all of the variables are unconstrained, i.e., variables may take on negative and positive values. Experimental results demonstrate reduction of systems in higher dimensions for both bounded and unbounded solution sets with feasible computational times, and provide important hindsight into its workings that allows us to design an extension of `ModFMEA` (`ExModFMEA`) that yields even more reduction.

Categories and Subject Descriptors

G.1.3 [**Mathematics of Computing**]: Numerical Analysis—*Numerical Linear Algebra*

General Terms

Symbolic transformation, convex hull, feasible solution, polytopes, linear inequalities, system reduction.

1. INTRODUCTION

The problem of eliminating redundancies in a system of linear inequalities arises in many applications. In linear programming one seeks a solution that optimizes a given objective function subject to a set of linear constraints, sometimes posed as linear inequalities. The *Simplex Method*, a method for finding the optima given a set of linear constraints, transforms these inequalities into a standard form [8]:

$$\text{minimize } \mathbf{cx}$$

$$\text{subject to } \mathbf{Ax} = \mathbf{b},$$

where $\mathbf{A}$ is the coefficient matrix in $\mathbb{R}^{M \times N}$, $\mathbf{x}$ is the parameter or variable column vector in $\mathbb{R}^N$, and $\mathbf{b}$ is the constant column vector in $\mathbb{R}^M$. The Simplex Method requires that the parameter vector be nonnegative. Inequalities are transformed into equalities by introducing auxiliary variables. Linear inequalities also arise in the context of tensor decomposition. Due to lack of uniqueness in higher-order tensor decomposition, nonnegativity constraints are imposed on the decomposition factors yielding systems of linear inequalities [2], [5]. Inevitably, these applications involve operations with systems of linear inequalities which are, at most times, redundant. Eliminating redundancies in such systems can reduce the number of computations, and hence improve computation times in applications. From the geometrical perspective, this problem is equivalent to finding the convex hull of an N-polytope expressed by a system of linear inequalities.

Current methods for eliminating redundant linear inequalities are not viable in higher dimensions [7]. They consist of applying a *redundancy test*, where a linear inequality is said to be redundant if replacing it with its negation results in an inconsistent system [7]. As an alternative we propose a modified version of the Fourier-Motzkin Elimination Algorithm (`ModFMEA`), implemented in MATLAB, to reduce redundancies in a given system of linear constraints over reals posed as linear inequalities, i.e., $\sum_{n=1}^{N} a_n x_n \geq b$ where a_n and b

are real constants. `ModFMEA` initially determines in what orthant(s) the solution set of the system lies in by intersecting all of the orthants that contain each half-space given by the linear inequalities. Reduction is obtained at each orthant by taking the lower and upper bounds of x_i-normalized inequalities $x_i \geq l$ and $u \geq x_i$ respectively, where l and u are linear terms with no occurrence of x_i, for $i = 1, 2, ..., N$. The reduced system over the whole solution set can be obtained by taking the union of the reduced system at each orthant.

This method eliminates redundancies by retaining a system of linear inequalities that define the set of feasible solutions. It works under the assumption that all of the variables are unconstrained, i.e., variables may take on negative and positive values. The algorithm effectively eliminates redundancies, in most cases total reduction, for system of linear inequalities in 2-D whose solution set is an unbounded convex cone. Experimental results also demonstrate reduction in systems of higher dimensions for both bounded and unbounded solution sets with feasible computational times. An extension of `ModFMEA` (`ExModFMEA`), that addresses the short comings of `ModFMEA`, is proposed at the end of the article.

The organization of the article is as follows. Section 2 covers the preliminary theoretical background for `ModFMEA`. Sections 3 and 4 describe the current `FMEA` and the proposed modified algorithm, `ModFMEA`. The testing strategy, results, and analysis are given in Section 5. Finally, the extension of `ModFMEA` is proposed in Section 6, with concluding remarks and future directions in Section 7.

2. PRELIMINARIES

A system of M linear inequalities in an N-dimensional Euclidian space can be expressed as a set of linear inequalities or by using matrices and vectors:

$$\mathbf{Ax} \geq \mathbf{b} \tag{1}$$

where $\mathbf{A}$ is the coefficient matrix in $\mathbb{R}^{M\times N}$, $\mathbf{x}$ is the parameter or variable column vector in $\mathbb{R}^N$, and $\mathbf{b}$ is the constant column vector in $\mathbb{R}^M$.

The solution set S of the system of linear inequalities $\mathcal{LI}$ is defined as follows:

$$S = \left\{\mathbf{x} \in \mathbb{R}^N | \mathbf{Ax} \geq \mathbf{b}\right\} \tag{2}$$

and denoted by $\mathcal{LI} \rightarrow S$. A similar definition can be extended to single linear inequalities.

Definition 1. **Equivalence:**
Consider the two systems of linear inequalities $\mathcal{LI}$ and $\mathcal{LI}'$. They are said to be *equivalent* if and only if they have the same solution set.

The definition of equivalence can naturally be extended to single inequalities.

There are three types of normal forms a linear inequality can take with respect to a given variable [1]. The following are definitions of the three possible norms with respect to the i^{th} variable for a linear inequality.

Definition 2. **Norms of linear inequalities:**
Consider the following linear inequality

$$\mathfrak{X} : \sum_{n=1}^{N} a_n x_n \geq b. \tag{3}$$

The ${}^{\geq}x_i$-norm, ${}^{\leq}x_i$-norm, and $\bar{x}_i$-norm of $\mathfrak{X}$ are linear inequalities defined as follows:

$$ {}^{\geq}x_i(\mathfrak{X}) : \sum_{n\neq i}^{N} \alpha_n x_n - \beta \geq x_i$$

$$ {}^{\leq}x_i(\mathfrak{X}) : x_i \geq \sum_{n\neq i}^{N} \alpha_n x_n - \beta$$

$$ \bar{x}_i(\mathfrak{X}) : \sum_{n\neq i}^{N} a_n x_n - b \geq 0$$

where $\alpha_n = -a_n/a_i$ and $\beta = -b/a_i$. The coefficient of the i^{th} variable must be negative, positive, and equal to zero for the ${}^{\geq}x_i$-norm, ${}^{\leq}x_i$-norm, and $\bar{x}_i$-norm respectively.

The following defines the relations $>$ and $<$ on $\mathcal{T}(\mathcal{N})$, the set of linear terms over N variables.

Definition 3. **Relations $>$ and $<$ on $\mathcal{T}(N)$:**
A linear term $t \in \mathcal{T}(N)$ is said to be less than (resp. greater than) $t' \in \mathcal{T}(N)$ if and only if for every $x_i \geq 0$, t is less than (resp. greater than) or equal to t' with the condition that $t \neq t'$, where $i = 1, 2, ..., N$:

$$t < t' \text{ (resp. } t > t' \text{) } \Leftrightarrow \forall x_1, x_2, ..., x_N \geq 0, t \neq t',$$

$$t(x_1, x_2, ..., x_N) \leq t'(x_1, x_2, ..., x_N).$$

$$\text{(resp. } t(x_1, x_2, ..., x_N) \geq t'(x_1, x_2, ..., x_N) \text{) }.$$

An important result of the $>$ and $<$ relations on $\mathcal{T}(N)$, and the foundation of our algorithm, is given by the following theorem.

THEOREM 1. *Let $\mathcal{LI}$ be a system of two linear inequalities: $\mathcal{LI} = \{x_i \geq t, x_i \geq t'\}$, where t,$t' \in \mathcal{T}(N-1)$ such that x_i does not occur in t and t'. Then*

$$t' > t \Leftrightarrow \mathcal{LI} \equiv x_i \geq t'.$$

Similarly, let $\mathcal{LI}'$ be another system of two linear inequalities: $\mathcal{LI}' = \{t \geq x_i, t' \geq x_i\}$. Then

$$t' < t \Leftrightarrow \mathcal{LI} \equiv x_i \leq t'.$$

PROOF. Consider the linear terms t and t' evaluated at some point such that $t = c$ and $t' = c'$, where $c, c' \in \mathbb{R}$. Then, by Definition 3

$$t' > t \Leftrightarrow c' > c$$

for all $x_n \geq 0$, where $n = 1, 2, ..., N$ and $n \neq i$. This implies that the solution set s' is contained in s, where $x_i \geq c' \rightarrow s'$ and $x_i \geq c \rightarrow s$, for $x_i \geq 0$. Since $\mathcal{LI} \rightarrow S$, it follows that

$$s' \subseteq s \Leftrightarrow S = s \cap s' = s' \Leftrightarrow \mathcal{LI} \equiv x_i \geq t'.$$

The proof for the second part of the theorem is omitted since it is very similar to the first. $\square$

The notion of minimum and maximum from the set of real numbers can be extended to the set of linear terms $\mathcal{T}(N)$ resulting in the following definitions.

Definition 4. **Minimum and maximum sets:**
Let T be a subset of $\mathcal{T}(N)$, a finite set of linear terms. The minimum set of T, denoted by $\min(T)$, is defined as

$$\min(T) = \{l \in T | \nexists t \in T : t < l\}. \tag{4}$$

Similarly, the maximum set of T, denoted by $\max(T)$, is defined as

$$\max(T) = \{u \in T | \nexists t \in T : t > u\}. \tag{5}$$

Definition 5. **Lower and upper bound inequalities:** Let T and T' be a subset of $\mathcal{T}(N-1)$ such that

$$T = \left\{t | t \geq x_i \in \ {}^{\geq}x_i(\mathcal{LI})\right\}$$

$$T' = \left\{t' | x_i \geq t' \in \ {}^{\leq}x_i(\mathcal{LI})\right\}$$

where t and t' are linear term with no occurrence of x_i. The set of lower and upper bound inequalities of $\mathcal{LI}$ with respect to x_i are denoted by L and U respectively, and defined as

$$U = \{u \geq x_i | u \in \min(T)\}$$

$$L = \{x_i \geq l | l \in \max(T')\}$$

where l and u are linear terms with no occurrence of x_i.

LEMMA 1. *Let ${}^{\geq}x_i(\mathcal{LI})$ be the ${}^{\geq}x_i$ norm of a system of linear inequalities $\mathcal{LI}$, and let U be the upper bound inequalities of $\mathcal{LI}$ with respect to x_i. Then*

$$U \equiv \ {}^{\geq}x_i(\mathcal{LI}).$$

LEMMA 2. *Let ${}^{\leq}x_i(\mathcal{LI})$ be the ${}^{\leq}x_i$ norm of a system of linear inequalities $\mathcal{LI}$, and let L be the upper bound inequalities of $\mathcal{LI}$ with respect to x_i. Then*

$$L \equiv \ {}^{\leq}x_i(\mathcal{LI})$$

THEOREM 2. *Let $\bar{x}_i(\mathcal{LI})$ be the $\bar{x}_i$ norm of a system of linear inequalities $\mathcal{LI}$, and let L and U be the lower and upper bound inequalities of $\mathcal{LI}$ with respect to x_i. Then*

$$U \cup L \cup \bar{x}_i(\mathcal{LI}) \equiv \mathcal{LI}.$$

PROOF. According to Lemmas 2 and 1,

$$U \equiv \ {}^{\geq}x_i(\mathcal{LI}) \text{ and } L \equiv \ {}^{\leq}x_i(\mathcal{LI}) \Leftrightarrow$$

$$U \cup L \cup \bar{x}_i(\mathcal{LI}) \equiv^{\geq} x_i(\mathcal{LI}) \cup^{\leq} x_i(\mathcal{LI}) \cup \bar{x}_i(\mathcal{LI})$$

$$\Leftrightarrow U \cup L \cup \bar{x}_i(\mathcal{LI}) \equiv \mathcal{LI}.$$

□

From Theorem 2 we can conclude that reducing the norm sets ${}^{\geq}x_i(\mathcal{LI})$ and ${}^{\leq}x_i(\mathcal{LI})$ to U and L respectively preserves the solution set of the original problem.

3. FOURIER-MOTZKIN ELIMINATION ALGORITHM

The Fourier-Motzkin Elimination Algorithm (FMEA) can be thought of as the Gaussian elimination algorithm for systems of linear inequalities. It consists of three steps:

1. The elimination step deletes variables from a given system of linear inequalities by applying their transitivity property, i.e. if $x \leq y$ and $y \leq z$ then $x \leq z$.
2. The deletion step removes any inequality of the form $a \geq b$, where a and b are constants and the inequality holds true.
3. Application of the failure step applies to contradictions (e.g., $-3 \geq 0$) in the system, where the program terminates concluding the system is inconsistent.

The `FMEA` applies the three steps repeatedly, terminating after an application of the failure step or when the system has been reduced to the empty set, implying consistency. The biggest draw back of the `FMEA` is its complexity. At each application of the elimination step the system expands exponentially, doubly exponentially to be precise [1]. Despite its caveats, the algorithm has proven of theoretical importance, in particular to the field of linear programming [6].

4. MODIFIED FMEA

A modified version of `FMEA` (`ModFMEA`) is implemented and developed in MATLAB as a method of eliminating redundancies in a given system of linear inequalities. The modification is made in the elimination step of the `FMEA` where redundant inequalities are eliminated instead of variables. Distinction between `FMEA` and `ModFMEA` is heightened by their functionality. `FMEA` checks for inconsistencies in a given system of linear inequalities while `ModFMEA` eliminates redundant inequalities. Another important difference is in their computational complexity. `ModFMEA` does not increase the size of the problem, i.e., the number of constraints does not increase as is the case in `FMEA`. A deeper analysis of `ModFMEA` complexity is provided at the end of the section.

`ModFMEA` consists of considering the trivial cases of redundancy first, i.e., eliminating inequalities that are positive multiples of another, and then analyzing the non-trivial cases. As a preprocessing step the `getQuadS` algorithm eliminates the orthants that do not contain any part of the solution set from the possible 2^N orthants for a system in N-D. The general idea for elimination in the non-trivial cases is to find the lower and upper bound inequalities of the system with respect to x_i using the `mini` function. The reduced system of linear inequalities is equivalent to the original problem constrained to the corresponding orthant. This is repeated for all variables and all orthants.The subsequent steps summarize `ModFMEA`.

Algorithm 1. **`ModFMEA`:**
For a given coefficient matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$, and a constant column vector $\mathbf{b} \in \mathbb{R}^M$, the following steps are performed to eliminate redundancies in the system of linear inequalities $\mathbf{Ax} \geq \mathbf{b}$, where $\mathbf{x} \in \mathbb{R}^N$.

```
1. Process trivial cases: eliminate linear inequalities
   that are positive multiples of others.
2. Apply getQuadS to obtain the orthants containing
   the solution.
3. Process non-trivial cases for each orthant:
   Loop for $k = 1, 2, ..., K$
   (a) Reduction with respect to each variable:
       Loop for $n = 1, 2, \dots, N$
         i. Normalize current $\mathcal{LI}$ with respect to $x_n$.
        ii. Determine L and U of $\mathcal{LI}$ with respect to
            $x_n$ applying the mini function.
       iii. Replace $\mathcal{LI}$ with $\mathcal{LI}'$ where
            $\mathcal{LI}' = L \cup U \cup \bar{x}_n(\mathcal{LI})$.
4. Return final $\mathcal{LI}$.
```

4.1 The getQuadS Algorithm

In the two dimensional rectangular coordinate space, the x and y axis divide the cartesian plane into 4 quadrants where the x and y coordinates are restricted to either negative or positive values. In general an N-dimensional Euclidian space is divided into 2^N *orthants*, quadrants for higher dimensional spaces. Orthants are represented here by row vectors whose entries take on the values of -1 or 1. Each component corresponds to a single coordinate, specifying its sign. For example, the second quadrant is represented by the row vector $q^{(2)} = [-1, 1]$. The set of all orthants from an N-dimensional Euclidian space is denoted by $\mathcal{Q}(N)$, a finite set with cardinality 2^N. For the case where $N = 2$,

$$\begin{aligned}\mathcal{Q}(2) &= \{[1,1], [-1,1], [1,-1], [-1,-1]\} \\ &= \left\{q^{(1)}, q^{(2)}, q^{(3)}, q^{(4)}\right\}.\end{aligned}$$

For a given system of linear inequalities of the form $\mathbf{Ax} \geq \mathbf{b}$ the `getQuadS`, algorithm deduces the set of orthants Q, ideally a subset of $\mathcal{Q}(N)$, containing the solution set. The algorithm essentially obtains a set of orthants for each linear inequality by analyzing the signs of both the coefficients and constants, taking their intersection to be Q.

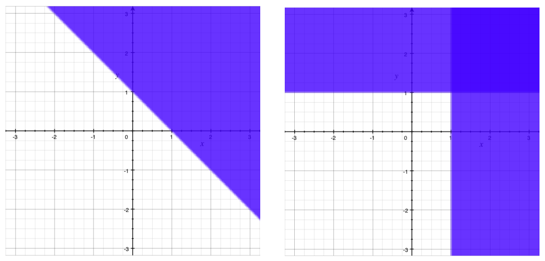
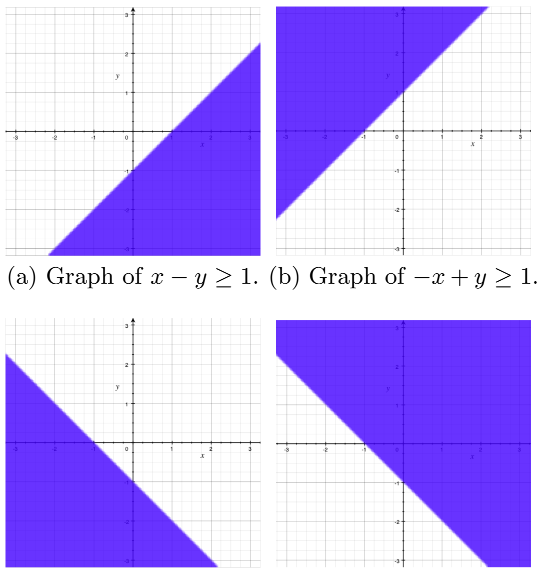

(a) Graph of $x + y \geq 1$. (b) Half-spaces with α and β equal to 1.

Figure 1: 2-D example.


(a) Graph of $x - y \geq 1$. (b) Graph of $-x + y \geq 1$.

(c) Graph of $-x - y \geq 1$. (d) Graph of $x + y \geq -1$.

Figure 2: 2-D example with alternative signs.

Consider the two-dimensional case with the inequality $a_1x + a_2y \geq b$, where $a_1 = 1, a_2 = 1$, and $b = 1$. From Fig. 1a we can deduce that the solution set is contained in three quadrants: $Q = \left\{q^{(1)}, q^{(2)}, q^{(4)}\right\}$. Q also corresponds to the union of the quadrants given by the two half-spaces in Eq. 6 and 7,

$$x \geq \alpha \tag{6}$$

$$y \geq \beta \tag{7}$$

where α and β are constants with the same sign as a_1 and a_2 respectively.

Changing the signs of a_1 and a_2 yields similar results, in the sense that the solution is still contained within three quadrants. As shown in Fig. 2a, 2b, and 2c changing the sign on a_1 results in a reflection of the graph about the y-axis, and vise versa. This also applies to the case where $b = 0$ and a_1 or a_2 are zero. When changing the sign on b, such that $b < 0$, results in a solution set contained by all of the quadrants, i.e., $Q = \mathcal{Q}(2)$ as shown in Fig. 2d. This is true for all possible combinations in sign of a_1 and a_2, and N-D inequalities. The following describes the `getQuadS` algorithm.

Algorithm 2. `getQuadS`:
For a given system of linear inequalities of the form $\mathbf{Ax} \geq \mathbf{b}$ with M inequalities and N parameters, the following steps are performed to acquire the set of orthants $Q \subseteq \mathcal{Q}(N)$ that contains the solution set of the system.

```
1. Loop for $m = 1, 2, ..., M$.
   Determine the set of orthants $Q_m$ for the $m^{th}$ inequality.
   (a) Check sign of $b_m$.
       If $b_m < 0$ return $\mathcal{Q}(N)$, else continue.
   (b) Loop for n = 1,2,...,N.
       Obtain $Q_m^{(n)}$, the set of orthants given by the half-space of the $n^{th}$ variable corresponding to the sign of $a_{mn}$.
       i. If $a_{mn} = 0$ return empty set.
   (c) Compute $\bigcup_{n=1}^{N} Q_m^{(n)} = Q_m$.
2. Compute and return $\bigcap_{m=1}^{M} Q_m = Q$.
```

4.2 Determining Lower and Upper Bound Inequalities

After processing the trivial cases and determining the set of orthants that contain the solution set, `ModFMEA` proceeds with the non-trivial cases. `ModFMEA` eliminates redundant inequalities by ascertaining the upper and lower bound inequalities with respect to each variable. Take the following example in 2-D where the system contains two inequalities:

$$\mathcal{LI} = \begin{cases} -y + x \geq 0 \\ -y + 3x \geq 0 \end{cases}.$$

The solution set is contained within quadrants I, III, and IV, as shown in Fig. 3. Consider first quadrant I where the second inequality is deemed redundant. Normalizing the system with respect to x yields the following:

$$\mathcal{LI} = \begin{cases} x \geq y \\ x \geq \frac{1}{3}y \end{cases}. \tag{8}$$

From the y coefficient it is deduced that the first inequality is indeed the lower bound inequality with respect to x. This implies that if y were to be evaluated for some $y \geq 0$, the lower bound inequality will yield a one-dimensional inequality that is contained inside the eliminated inequality. For example, take $y = 3$. This results in $x \geq 3$ and $x \geq 1$

from Eq. 8, where it is obvious that the second inequality is redundant. Determining lower and upper bound inequalities in 3-D can be visualized in a similar way. Take two linear inequalities in 3-D constrained to the all-positive octant, where $x, y, z \geq 0$. Fixing the variable z at some constant value, equivalent to obtaining a 2-D slice of the volume through the z-axis, will result in a 2-D graph similar to Fig. 3. Fixing either x or y, taking a 1-D slice, will yield one-dimensional inequalities. If the first inequality were to be the lower bound then it would imply that all 1-D slices, from any 2-D slice, will result in a 1-D inequality contained within the redundant inequality provided that the slices are made for $x, y, z \geq 0$. A similar process follows for upper bounds and higher dimensions.

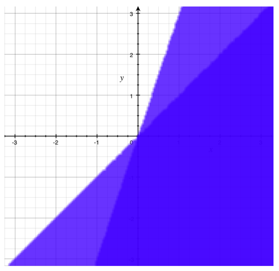


Figure 3: 2-D example of non-trivial cases.

From Definition 4.2 the lower and upper bound inequalities are systematically determined by obtaining the maximum and minimum sets of the corresponding linear terms. According to Theorem 2, this is done by comparing the coefficients and constants of the linear inequalities. The following is an algorithm implemented in MATLAB as a function, used to find the upper bound inequalities from a given $^{\leq}x_i(\mathcal{LI})$-norm set.

```
Algorithm 3. mini:
Given a set of linear terms, the mini function returns its minimum set.
  1. Initialize $i$ to the number of linear terms.
  2. While $i > 1$
     (a) Set $k = i - 1$ and the reference term to the $i^{th}$ linear term.
     (b) Loop for $j = k, k-1, k-2, \ldots, 1$
         i. Compare the reference term with the $j^{th}$ linear term.
         ii. If reference term is lesser
             - delete $j^{th}$ linear term
             - update $i = i - 1$
         iii. Else if $j^{th}$ linear term is lesser
             - delete $i^{th}$ linear term
             - and break loop
     (c) Update $i = i - 1$
  3. Return minimum set.
```

The `mini` function is also used to determine the maximum set. For a given set of linear terms T, the maximum of the set is equal to the minimum set of $-T$.

4.3 Computational Complexity

Referring back to Algorithm 3, we can see that up to $\frac{M(M+1)}{2}$ comparisons between linear terms are made in the case where no reduction is possible on a set of M linear terms. However, since comparing two linear terms originating from an N-D system results in N comparisons, the `mini` function makes a total of $\frac{NM(M+1)}{2}$ comparisons. An implementation similar to `mini` is developed for processing trivial cases, therefore its worst-case complexity is $\mathcal{O}(NM^2)$. The `getQuadS` algorithm, as described in Algorithm 2, generates NM orthants, where each orthant takes $2^{\frac{N(N+1)}{2}}$ operations, yielding a worst-case complexity of $\mathcal{O}(2^{\frac{N(N+1)}{2}}NM)$.

From Algorithm 1 we arrive at an overall estimate of `ModFMEA`'s worst case complexity for a system of linear inequalities with N variables and M inequalities. The function `mini` is called a total of $2N$ times per orthant, for a total of $2^{(N+1)}N$ times in the worst case, while the trivial cases and the `getQuadS` algorithm are processed once. This results in the following total number of comparisons:

$$\left[\frac{M(M+1)}{2}\right] + 2^{\frac{N(N+1)}{2}}NM + 2^{(N+1)}N\left[\frac{M(M+1)}{2}\right]$$
$$= \mathcal{O}(2^N N^2 M^2). \qquad (9)$$

As shown by Eq. 9, `ModFMEA` is a more computationally feasible algorithm than the doubly exponential `FMEA`. This is mainly due to the fact that `ModFMEA` maintains the size of the problem constant in the worst case where no reduction is possible. In the cases where reduction is achieved, the number of operations decreases as eliminations are made.

5. TESTING

The algorithm is tested on its effectiveness in eliminating redundant inequalities considering the following:

- dimensional space of the system;
- whether the solution set is bounded or unbounded;
- size of the overall system: total number of inequalities;
- geometry and location of solution sets: what orthant(s) contain the solution set;
- and *relative core size*: the ratio between the number of non-redundant and total number of inequalities.

Consistent systems of linear inequalities of the form $\mathbf{Ax} \geq \mathbf{b}$ are generated for test cases. These systems possess feasible solution sets that are either bounded or unbounded on a varying multitude of Euclidian dimensional spaces.

5.1 Description of Testing Strategy

Bounded solution sets of systems of linear inequalities are geometrically described as bounded convex polytopes. The linear inequalities that define this polytope, i.e., the non-redundant inequalities, will be referred to as the *core*. Generating the feasible system of linear inequalities consists of first generating the core, which can be specified to have at least $N+1$ inequalities for an N-D space. The core is randomly generated such that it is confined by an N-hypercube centered at the origin with length 50. In other words, the coordinates of each vertex of the polytope are randomly generated within the interval $[-25, 25]$. As of consequence the solution sets acquire a sufficiently random geometry at random locations, allowing for the testing of different geometries that could occupy any feasible combination of orthants. The redundant inequalities are then generated such that they contain, but do not intersect, the core. This method allows us control over the number of redundant, non-redundant, and total number of linear inequalities in our system.

Systems with **unbounded solution sets** are analogously geometrically described by unbounded convex-polytopes. However, for simplicity, only systems whose solution set results in a convex cone are generated. This is the case for systems of the form $\mathbf{Ax} \geq \mathbf{0}$, common in tensor decomposition and other applications. These systems are generated with the same approach described for systems with bounded solution sets. With the exception of the second dimension, the number of non-redundant inequalities is for simplicity limited to at least N inequalities for an N-D space. When dealing with convex cones in 2-D, the number of non-redundant inequalities is fixed to two.

Test cases were generated for all dimensions between 2 and 10, for a total of 9 dimensions tested. Three different relative core sizes were generated for each dimension: 10%, 20%, and 30%. A total of 20 systems were generated for each relative core size, per dimension, for a total of $20\times3\times9 = 540$ systems of varying sizes for each bounded and unbounded cases (a total of 1080 systems were generated). Since there are constraints on the minimum number of non-redundant inequalities, the size of the system is dependent on the number of inequalities that represent the smallest relative core size: 10%. For consistency between the bounded and unbounded cases, this number is chosen to be $N+1$, where N is the dimension, and hence the total number of inequalities generated for a system in N-D is $(N+1)\times 10$. For example, the 3-D systems have 40 inequalities, where the number of non-redundant inequalities goes from 4, 8, and 12 corresponding to the three relative core sizes. As the exception, unbounded cases for 2-D systems has only 30 inequalities with 2 non-redundant inequalities.

5.2 Results

The results are summarized by Tables 1-2, and Fig. 4-5 for both bounded and unbounded cases respectively. The maximum and minimum number of reductions made, or the number of inequalities eliminated, shown in Tables 1 and 2 is acquired from all three relative core sizes. The size of the core refers to the number of non-redundant inequalities for the respective dimension, and varies according to the relative core size. For example, the core size in Table 1 for 3-D polyhedrons varies between 4 and 12, specifically 4,8, and 12 for the corresponding relative core sizes. According to Table 1 the maximum number of reductions observed for generated polyhedra, in all relative core sizes, is 32 from a total system size of 40 where the number of non-redundant inequalities varied from 4, 8, and 12.

Fig. 4 and Fig. 5 are graphs of relative average reduction, average number of reductions divided by the number of redundant inequalities, at each relative core size versus dimension. The magenta line with triangle markers in Fig. 5 shows that on average, the amount of reduction at 30% relative core size dropped from nearly 100% in 2-D to 30% in 10-D. In other words, the algorithm was able to reduce approximately a third of the redundant inequalities in 10-D whose system contained 30% non-redundant inequalities.

Dimension	Size of System	Size of Core	Maximum # of reductions	Minimum # of reductions
2	30	3-9	20	4
3	40	4-12	25	0
4	50	5-15	19	0
5	60	6-18	12	0
6	70	7-21	23	0
7	80	8-24	31	0
8	90	9-27	74	0
9	100	10-30	29	0
10	110	11-33	68	0

Table 1: Bounded cases.

Dimension	Size of System	Size of Core	Maximum # of reductions	Minimum # of reductions
2	30	2-2	28	26
3	40	4-12	27	4
4	50	5-15	28	0
5	60	6-18	48	0
6	70	7-21	35	0
7	80	8-23	61	0
8	90	9-27	47	0
9	100	10-30	73	0
10	110	11-33	60	0

Table 2: Unbounded cases.

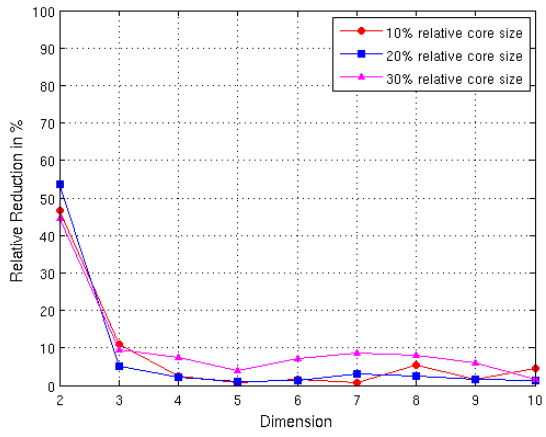


Figure 4: Relative average reduction for bounded cases.

5.3 Analysis

`ModFMEA` is shown to eliminate redundancies for both bounded and unbounded cases for several relative core sizes. Results demonstrated overall cases of under-reduction, where `ModFMEA` was unable to eliminate all redundant inequalities. Examining `ModFMEA` reveals two main causes of under-reduction: undetectable redundant inequalities, and empty orthants.

Undetectable redundant inequalities are inequalities that despite being redundant `ModFMEA` is unable to eliminate them since they are perceived as non-redundant. The cause of these cases can be traced back to the definition of the "$>$" and "$<$" relations on linear terms. When eliminating redundancies `ModFMEA` compares the corresponding inequalities over the interval $[0,\infty)$ and/or $(-\infty,0]$ depending on

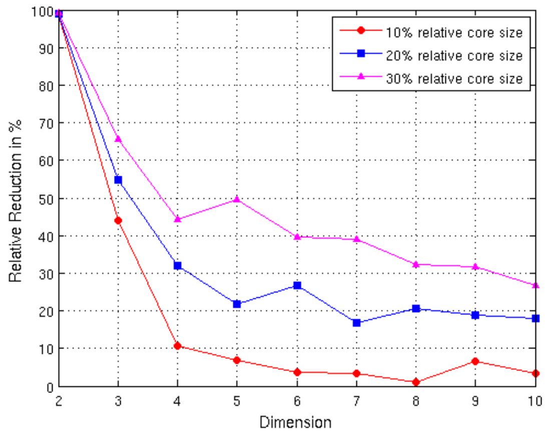


Figure 5: Relative average reduction for unbounded cases.

the orthant. If two linear inequalities intersect within the orthant where elimination is taking place, neither inequality can be eliminated. Hence, a redundant inequality becomes undetectable if it intersects all of the corresponding non-redundant inequalities of the orthant in question. This leads to the accumulation of undetectable redundant inequalities as reduction takes place at each orthant, from which the union of the reduced system is taken according to Algorithm 1.

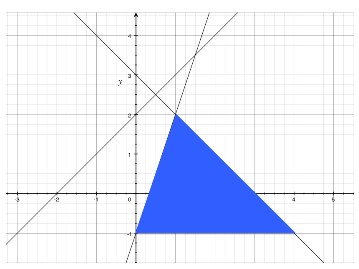
Consider the following example:

$$\mathcal{LI} = \begin{cases} -y + x \geq -2 \\ -y + 3x \geq 1 \\ -y - x \geq -3 \\ y \geq -1 \end{cases} . \quad (10)$$

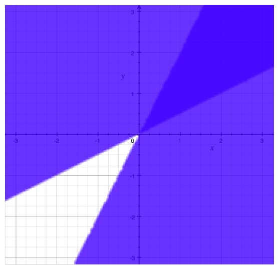
where the solution set is graphically shown by the shaded area in Fig. 6a. The redundant inequality is given by the first inequality in Eq. 10. Reduction with respect to the first quadrant should yield the second and third inequalities as the non-redundant inequalities. However, application of `ModFMEA` results in eliminating only the last inequality. Since the redundant inequality intersects both of the non-redundant inequalities in the first quadrant it becomes an undetectable redundant inequality, even if the intersection is outside of the solution set.

Empty orthants refer to orthants that do not contain any part of the solution. Applying `ModFMEA` at such orthants leads to improper reduction. The `getQuadS` algorithm generates these empty orthants due to its inability to completely eliminate them. The set of orthants returned by the algorithm does however contain the solution set, so no information is lost. The following system, given in Fig. 6b, demonstrates a case of empty orthant retention. Clearly it can be deduced that the first quadrant is the only quadrant that contains the solution set. The first step in `getQuadS` algorithm is to obtain the quadrants that contain the solution set of each inequality. For this example, set of quadrants for one of the inequalities includes the 1^{st}, 2^{nd} and 3^{rd} quadrants. The other inequality's solution set is contained within the 1^{st}, 3^{rd}, and 4^{th} quadrants. Taking the intersection between the two sets yields the 1^{st} and 3^{rd} quadrants, where the latter is an empty quadrant.

The relative average reduction for both bounded and unbounded cases reveals a negative correlation between the effectiveness of `ModFMEA` in eliminating redundant inequalities and higher dimensions (see Fig. 4 and 5). This is attributed to the fact that empty orthant retention and cases of undetectable redundant inequalities increase with higher dimensions. The more dimensions there are the easier it is for a redundant inequality to intersect the non-redundant inequality. Also, higher dimensions lead to an increase number of orthants from which undetectable redundant inequalities may originate, and an increase in the number of retained empty orthants.



(a) Undetectable redundant inequality.



(b) Empty orthant retention.

Figure 6: 2-D examples of causes of under-reduction.

6. EXTENSION OF MODFMEA

The notion of lower and upper bound inequalities in `ModFMEA` offers a condition for eliminating redundant inequalities. However, this condition is rather strict in the sense that the inequalities in question are compared for all $x_i \geq 0$, as defined in the "$>$" and "$<$" relations on the set of linear terms. Despite its shortcomings, `ModFMEA` has laid the theoretical groundwork for a more efficient method, the extended `ModFMEA` (`ExModFMEA`). Instead of requiring the upper and lower bound inequalities be defined over the interval $[0, \infty)$ for each variable, or $(-\infty, 0]$ depending on the orthant, `ExModFMEA` defines these bounds over the intervals that the parameters are constrained to according to their solution set. Obtaining the bounds for the solution set can be achieved through interval solvers [4].

Bounded solution sets are given by bounded convex N-polytopes whose parameters are constrained by finite intervals , i.e. the N-polytope can be enclosed by a N-hyperrectangle or N-orthotope (the N-D generalization of rectangles). It is sufficient to analyze the inequalities at these end points. Similarly, unbounded solution sets are given by unbounded convex N-polytopes, or convex cones in the case where $\mathbf{b} = \mathbf{0}$. In this case all of the parameters can be artificially bounded by replacing the occurrences of $-\infty$ and ∞, for simplicity, with -1 and 1 respectively. This would allow for further reduction of redundant inequalities as described for the bounded cases.

The following illustrates the application of `ExModFMEA` to system given by Eq. 10. From Fig. 6a we can deduce that x and y are bounded to the intervals $[0, 4]$ and $[-1, 2]$ respectively. It is important to mention that in practice, interval solvers will not yield intervals as tight as the ones given in this example. This is mainly due to the fact that interval solvers provide outer approximations, which most often are wider intervals (depending on the geometry of the system) that are still useful for reduction [3], [4]. Transforming Eq. 10 to the x-norm yields the following system:

$$\begin{cases} x \geq y - 2 \\ x \geq \frac{1}{3}y + \frac{1}{3} \\ -y + 3 \geq x \\ y \geq -1 \end{cases} . \tag{11}$$

The last two inequalities correspond to the $^{\geq}x(\mathcal{LI})$ and $\bar{x}(\mathcal{LI})$ norms where no elimination is possible since each norm contains a single inequality. Evaluating the $^{\leq}x(\mathcal{LI})$-norm inequalities for y at the lower and upper bounds results in four 1-D inequalities. This is equivalent to taking a 1-D slice of the solution, parallel to the x-axis, for $y = -1$ and $y = 2$ as shown in Fig. 7.

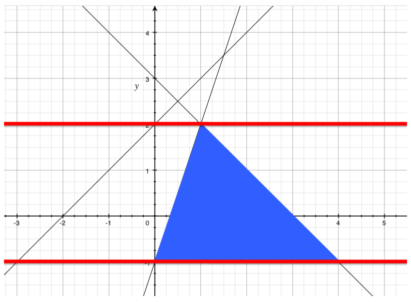


Figure 7: Solutions set bounded by $y \in [-1, 2]$.

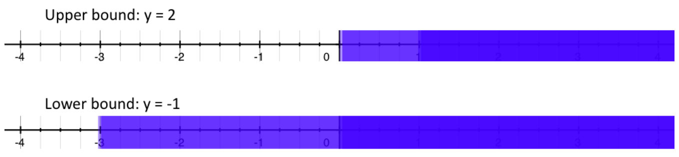


Figure 8: Resulting 1-D inequalities.

The inequality $x \geq y - 2$ generates $x \geq -3$ and $x \geq 0$ when evaluating y at its lower and upper bounds respectively. Similarly $x \geq \frac{1}{3}y + \frac{1}{3}$ yields the following inequalities: $x \geq 0$ and $x \geq 1$. Comparing the upper and lower bound generated inequalities separately as done in Fig. 8 reveals thats the 1-D inequalities resulting from $x \geq \frac{1}{3}y + \frac{1}{3}$ are contained inside the inequalities given by $x \geq y - 2$, and hence we can deduce that $x \geq y - 2$ is redundant.

7. CONCLUSION

Systems of linear inequalities arise in many applications where the computational complexity of operations involving such systems is dependent on the total number of inequalities. Eliminating redundancies in such systems can reduce the number of computations, and hence improve computation times in applications. Current techniques in for eliminating redundant inequalities are not viable in higher dimensions [7].

We propose a modified version of the `FMEA,` a more geometrical approach for eliminating redundancies. `ModFMEA` eliminates redundancies by first processing trivial cases, where one inequality is a positive multiple of another. The algorithm then eliminates non-trivial redundancies by determining the lower and upper bounds of x_i-normalized inequalities at each orthant containing the solution set. A total of 1080 test cases were generated for in a variety of dimensions ranging from 2-D to 10-D. Experimental results demonstrate reduction for bounded and unbounded cases in all dimensions tested with feasible computational times. Results also affirm a negative correlation between efficiency in reduction and the increase of dimension. Further analysis of `ModFMEA` reveals susceptibility to cases of under-reduction due to undetectable redundant inequalities and elimination in empty orthants.

Despite its shortcomings, `ModFMEA` has provided the theoretical groundwork for a more efficient method of eliminating redundancies. We propose an extension of `ModFMEA` (`ExModFMEA`) which specifically addresses the issues of under-reduction. A bound is approximated for each variable using an interval solver, from which lower and upper bound inequalities with respect to x_i are determined by analyzing the resulting inequalities at the boundaries of the solution set. This process also applied to unbounded cases. Future directions include implementation and extensive testing of `ExModMEA`.

8. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0953339.

9. REFERENCES

[1] K. Apt. *Principles of constraint programming.* Cambridge Univ Press, 2003.

[2] L. De Lathauwer. A survey of tensor methods. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pages 2773–2776. IEEE, 2009.

[3] L. Granvilliers. On the combination of interval constraint solvers. *Reliable Computing*, 7(6):467–483, 2001.

[4] L. Granvilliers. RealPaver user's manual. *IRIN, University of Nantes, 0.1 edition*, 2002.

[5] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[6] J. Lassez and M. Maher. On Fourier's algorithm for linear arithmetic constraints. *Journal of Automated Reasoning*, 9(3):373–379, 1992.

[7] M. Le Fur. Scanning parameterized polyhedron using Fourier-Motzkin elimination. *Concurrency: Practice and Experience*, 8(6):445–460, 1996.

[8] D. Luenberger. *Introduction to Linear and Nonlinear Programming.* Addison-Wesley Company Inc., 1973.