2014-01-01

Fast Algorithm For Finding Lattice Subspaces In Rn And Its Implementation

Andrew Martin Pownuk
University of Texas at El Paso, ampownuk@utep.edu

FAST ALGORITHM FOR FINDING LATTICE SUBSPACES IN $\mathbb{R}^n$ AND ITS IMPLEMENTATION

ANDREW M. POWNUK

Department of Mathematical Sciences

APPROVED:

Piotr Wojciechowski, Ph.D., Chair

Maria Mariani, Ph.D.

Vladik Kreinovich, Ph.D.

Charles Ambler, Ph.D.

Dean of the Graduate School

FAST ALGORITHM FOR FINDING LATTICE SUBSPACES IN $\mathbb{R}^n$ AND ITS IMPLEMENTATION

by

ANDREW M. POWNUK

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Mathematical Sciences

THE UNIVERSITY OF TEXAS AT EL PASO

December 2014

Abstract

There are known necessary and sufficient conditions for a subspace of $\mathbb{R}^m$ to be lattice-ordered. Let $Y = \{y_1,...,y_m\}$ and y_i are rows of the matrix X. A subspace $\langle X \rangle$, of linear space generated by the set X of *n* linearly independent positive vectors is lattice-ordered if and only the set X admits a fundamental set of indices I, which means that the subset $Y_I \subseteq Y$ of vectors indexed by I is linearly independent, and every vector from $Y \setminus Y_I$ is a nonnegative linear combination of vectors form Y_I.

In economics it is possible to prove that the minimum-cost insured portfolio exists if and only if the linear space generated by the corresponding financial instruments is lattice-ordered.

In the literature there are known algorithms with exponential complexity that determine if a given subspace is lattice-ordered.

In this thesis a polynomial time algorithm (serial and parallel) as well as its computer implementation will be presented. The method can be applied in economics as well as in the theory of vector lattices.

Table of Contents

List of Tables

List of Figures

Notation

$a \vee b$ - a join, 6

$a \wedge b$ - a meet, 6

$W^{\perp}$ - an orthogonal complement of the subspace W, 12

$\dim V$ - dimension of the linear space, 10

$\bigvee S$ - join - smallest upper bound of the set S, 5

$\bigwedge S$ - meet - greatest lower bound of the set S, 5

V^{+} - positive cone, 9

0_P - the bottom element of P, 3

$\left[v, w\right]$ - the closed line segment, 11

$\mathbb{F}$ - the field, 7

$x\left(i\right)$ - the *i-th* component of x, 13

$\mathrm{length}\left(\{a, c, g\}\right)$ - the *length of the chain* $\{a, c, g\} \subseteq P\{a, c, g\} \subseteq P$, 3

$\left(V, \mathbb{F}, \circ, +\right)$ - the linear space, 8

$\mathrm{ray}\left(v\right)$ - the ray generated by a vector $v \neq 0$, 11

S^{l} - the set of all lower bounds of the set S, 4

$\mathbb{R}$ - the set of all real numbers, 8

S^{u} - the set of all upper bounds of the set S, 4

1_P - the top element of P, 4

$\mathrm{width}\left(P\right)$ - the width of a partially ordered set $\left(P, \leq\right)$, 3

$\left|x\right|$ - modulus of *x*, 9

x^{-} -negative part of *x*, 9

x^{+} -positive part of *x*, 9

GLB(*S*) – the greatest lower bound of the set *S*, 5

GreatestElement(*S*) – the greatest element of the set *S*, 4

.

GreatestLowerBound(*S*) – the greatest lower bound of the set *S*, 5

L - the number of the bits in the input, 25

LB(*S*) – the lower bound of the set *S*, 4

LeastElement(*S*) - the least elements in the set *S*, 3

LeastUpperBound(*S*) – the least upper bound of the set *S*, 5

LowerBound(*S*) – the lower bound of the set *S*, 4

LUB(*S*) – the least upper bound of the set *S*, 5

MaximalElement(*S*) - the set of maximal elements in the set *S*, 3

MinimalElement(*S*) - the set of minimal elements in the set *S*, 3

UB(*S*) – the upper bound of the set *S*, 4

UpperBound(*S*) – the upper bound of the set *S*, 4

Chapter 1: Partially Ordered Sets

In this section basic information about partial ordered sets and their properties will be presented.

Definition 1 - *Partially ordered set* is a pair $\left(P,\leq\right)$ where P is a set and $\leq$ is a relation such that:

1) $a \leq a$ (reflexivity),

2) if $a \leq b$ and $b \leq a$ then $a = b$ (antisymmetry),

3) if $a \leq b$ and $b \leq c$ then $a \leq c$ (transitivity).

Example 1 - A pair $\left(\mathbb{N},|\right)$ where $\mathbb{N}$ is the set of natural and $|$ is the relation of divisibility is the partially ordered set and for example

$$2 \leq 6 \Leftrightarrow 2 \mid 6 \Leftrightarrow 6 = 3 \cdot 2 \text{ and } 3 \in \mathbb{N}. \tag{1}$$

Example 2 - A pair $\left(\mathbb{R}^2,\leq_{\mathbb{R}^2}\right)$ is an example of the partial set and for example

$$\left(1,2\right) \leq_{\mathbb{R}^2} \left(3,5\right) \Leftrightarrow \left(1 \leq 3\right) \text{and} \left(2 \leq 5\right). \tag{2}$$

Definition 2 - Let us consider partially ordered set $\left(P,\leq\right)$. If $x \leq y$ and $x \neq y$ then it is possible to write $x < y$ where $<$ is the *strict partial order* associated to $\leq$.

Definition 3 - If $\left(a \leq b\right) \vee \left(b \leq a\right)$ then a,b are *comparable*.

Example 3 - $\left(1,2\right)$ and $\left(2,3\right)$ are comparable in the poset $\left(\mathbb{R}^2,\leq_{\mathbb{R}^2}\right)$ because $1 \leq 2$ and $2 \leq 3$.

Definition 4 - Let $\left(P,\leq_P\right)$ be a poset on a set X (i.e. $P \subseteq X$), and $A \subseteq X$ We define a suborder $\leq_A$ as the restriction of $\leq_P$ to the pairs of A:

$x \leq_A y \Leftrightarrow \left(x \leq_P y\right) \wedge \left(x,y \in A\right)$.

Then $\left(A,\leq_A\right)$ is a *subposet* of P.

Definition 5 - If $\neg\left(a \leq b\right)$ and $\neg\left(b \leq a\right)$ then a,b are *incomparable*.

Example 4 - $(2,3)$ and $(4,1)$ are incomparable in the poset $(\mathbb{R}^2, \leq_{\mathbb{R}^2})$ because $2 \leq 4$ and $3 \geq 1$.

Definition 6 - Let us consider a poset $(P, \leq)$. When P is finite, one can represent poset conveniently using the covering relation (which determines $\leq$).

In the *Hasse diagram* of P if $x < y$ then we draw a line connecting x and y if x is an immediate predecessor of y (i.e. y covers x).

Example 5 - Let X be a set, and consider the family of 2^X of its subsets with respect to inclusion. Then $(2^X, \subseteq)$ is a poset that is called the subset poset on X.

Let us consider the finite set $X = \{1,2,3\}$, the Hasse diagram of 2^X is given in the Fig. 1.

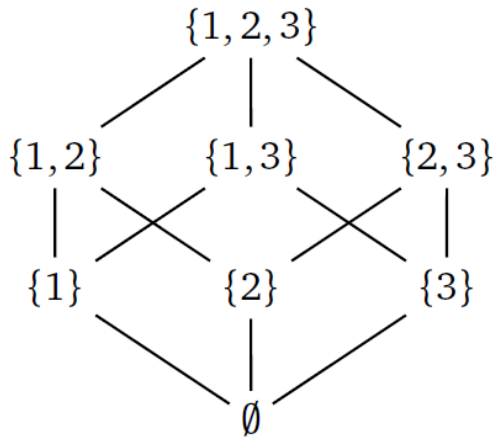

Figure 1: Poset from the example 5.

Definition 7 - Let us consider a partially ordered set $(P, \leq)$. A subset S of P is called a *chain* if every two elements in the subset are comparable.

Definition 8 - A chain A in P is *maximal* if for all $x \in P \setminus A$, $A \cup \{x\}$ is not a chain.

Definition 9 - Let us consider a partially ordered set $(P, \leq)$. A subset S of P is called an *antichain* if every two elements in the subset are incomparable.

Definition 10 - An antichain A of P is *maximal* if for all $x \in P$, x is comparable to an element in A (including the possibility that $x \in A$).

Definition 11 - The *width* of a partially ordered set $\left(P, \leq\right)$, denoted $\text{width}\left(P\right)$, is the maximal size of an antichain in P.

Definition 12 - We refer to the number of elements in a chain as the *length* of the chain.

Definition 13 - A pair $\left(P, \leq\right)$ is *totally ordered* if exists relation $\leq$ such that:

1) $a \leq a$ (reflexivity),

2) if $a \leq b$ and $b \leq a$ then $a = b$ (antisymmetry),

3) if $a \leq b$ and $b \leq c$ then $a \leq c$ (transitivity),

4) all $a, b \in P$ are comparable i.e. $\forall a, b \in P, \left(a \geq b\right) \text{or} \left(b \geq a\right)$.

Definition 14 - An *interval* in a partially ordered set can be defined as $\left[x, y\right] = \left\{z \in P : x \leq z \wedge z \leq y\right\}$. If $\neg\left(x \leq y\right)$ then $\left[x, y\right] = \varnothing$.

Definition 15 - A poset $\left(P, \leq\right)$ is said to be *locally finite* if every interval $\left[x, y\right] \subseteq P$ is finite.

Definition 16 - Let us consider a partially ordered set $\left(P, \leq\right)$. An element x in P is called a *minimal element* if there is no $y \in P$ such that $y \leq x$.

Elements of the set MinimalElement(*S*) don't have to be comparable.

Definition 17 - Let us consider a partially ordered set $\left(P, \leq\right)$.

An element x in P is called a *least element* if for all $y \in P$, $x \leq y$.

The least element has to be comparable with all elements in the set S.

Definition 18 - Let us consider a partially ordered set $\left(P, \leq\right)$.

An element x in P is called a *maximal element* if there is no $y \in P$ such that $x \leq y$.

Maximal elements of the set don't have to be comparable.

Definition 19 - Let us consider a partially ordered set $\left(P,\leq\right)$.

An element x in P is called a *greatest element* if for every $y \in P$, $x \leq y$.

Greatest element has to be comparable with all elements in the set S.

Theorem 1 - A greatest and/or least element of a poset may or may not exist, but is unique if it does.

Example 6 - Let us consider the poset $\left(P,|\right)$ where P={1,2,3,6,8,12} then MinimalElement(P)={1} and MaximalElement(L)={8,12}. The set P has no greatest element (i.e. GreatestElement(P)=Ø) and 1 is the least element of the set P i.e. LeastElement(L)=1.

Example 7 - Let us consider the poset $\left(L,\subseteq\right)$ where $L = P\left(\left\{1,2,3\right\}\right) = 2^{\{1,2,3\}}$, then GreatestElement($L$)=MinimalElement($L$)=Ø and LeastElement($L$)=MaximalElement($L$)={1,2,3}.

Example 8 - $\left(\mathbb{Z},\leq\right)$ has no maximal elements nor any minimal elements where $\mathbb{Z} = \left\{\ldots,-2,-1,0,1,2,\ldots\right\}$.

Definition 20 - For a partially ordered set $\left(P,\leq\right)$ a *lower bound* $x \in P$ of some subset $S \subseteq P$ is an element of P which is smaller or equal to every element of S i.e. $\forall y \in S, x \leq y$.

Lower bound have to be comparable with all elements in the set S.

The set of all lower bounds of the set S can be denoted as $S^l = \left\{x \in P : \forall y \in S, x \leq y\right\}$.

If $S = P$ then a lower bound is the same as a least element.

Definition 21 - For a partially ordered set $\left(P,\leq\right)$ an *upper bound* $x \in P$ of some subset $S \subseteq P$ is an element of P which is bigger or equal to every element of S i.e. $\forall y \in S, y \leq x$.

Upper bound have to be comparable with all elements in the set S.

The set of all upper bounds of the set S can be denoted as S^u $S^u = \left\{x \in P : \forall y \in S, y \leq x\right\}$.

If $S = P$ then an upper bound is the same as a greatest element.

Definition 22 - Let us consider a partially ordered set $\left(P,\leq\right)$, some subset $S \subseteq P$, and a set of upper bounds of the set S i.e. $S^u = \left\{x \in P : \forall y \in S, y \leq x\right\}$. If there exists some $u \in S^u$ such that for all $v \in S^u$, $u \leq v$, then u is also called a *least upper bound* of S.

The least upper bound is denoted as $\bigvee S$ and is called a *join* or *supremum* of S, and denoted by $\sup S$.

$$\bigvee S = \text{sup } S = \text{LestElement(UpperBound(}S\text{))} \tag{3}$$

Definition 23 - Let us consider a partially ordered set $\left(P,\leq\right)$, some subset $S \subseteq P$, and a set of lower bounds of the set S i.e. $S^l = \left\{x \in P : \forall y \in S, x \leq y\right\}$. If exists some $u \in S^l$ such that for all $v \in S^l$, $v \leq u$, then u is called a *greatest lower bound* of S.

The greatest lower bound is denoted as $\bigwedge S$ and is called a *meet* or *infimum* of S, and denoted by $\inf S$.

$$\bigwedge S = \text{inf } S = \text{GreatestElement(LowerBound(}S\text{))} \tag{4}$$

Example 9 - Let us consider the poset $\left(2^{\{1,2,3\}},\subseteq\right)$ (compare the example 5) and $S = \left\{\left\{1\right\},\left\{1,2\right\}\right\}$.

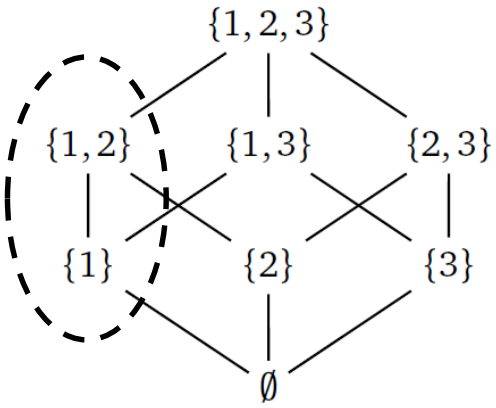

Figure 2: Set $S = \left\{\left\{1\right\},\left\{1,2\right\}\right\}$.

We know that

$$\text{LowerBound}\left(S\right) = \text{LowerBound}\left(\left\{\left\{1\right\},\left\{1,2\right\}\right\}\right) = \left\{\varnothing,\left\{1\right\}\right\} \tag{5}$$

$$\mathrm{UpperBound}\left(S\right) = \mathrm{UpperBound}\left(\left\{\left\{1\right\},\left\{1,2\right\}\right\}\right) = \left\{\left\{1,2\right\},\left\{1,2,3\right\}\right\} \tag{6}$$

then

$$\bigwedge S = \inf S = \mathrm{GreatestElement}\left(\mathrm{LowerBound}\left(S\right)\right) = \left\{1\right\} \tag{7}$$

$$\bigvee S = \sup S = \mathrm{LestElement}\left(\mathrm{UpperBound}\left(S\right)\right) = \left\{1,2\right\} \tag{8}$$

Definition 24 - A partially ordered set $\left(L,\leq\right)$ is called a *meet-semilattice* if for each two-element subset $\left\{a,b\right\} \subseteq L$ has a meet (i.e. greatest lower bound), denoted $a \wedge b$.

Definition 25 - A partially ordered set $\left(L,\leq\right)$ is called a *join-semilattice* if for each two-element subset $\left\{a,b\right\} \subseteq L$ has a join (i.e. least upper bound), denoted $a \vee b$.

Definition 26 - A *lattice* is a partially ordered set in which every two elements have a least upper bound and also called a greatest lower bound.

Example 10 - Partially ordered set $\left(2^{\{1,2,3\}},\subseteq\right)$ is a lattice (compare Fig. 1) because every pair of the element has a least upper bound and the greatest lower bound

$$\varnothing \wedge \left\{1\right\} = \varnothing \in 2^{\{1,2,3\}}, \varnothing \vee \left\{1\right\} = \left\{1\right\} \in 2^{\{1,2,3\}} \tag{9}$$

$$\left\{1\right\} \wedge \left\{2\right\} = \varnothing \in 2^{\{1,2,3\}}, \left\{1\right\} \vee \left\{2\right\} = \left\{1,2\right\} \in 2^{\{1,2,3\}} \tag{10}$$

etc.

Definition 27 - A *complete lattice* is a partially ordered set $\left(L,\leq\right)$ in which all subsets have both a supremum, $\bigvee A$, and an infimum, $\bigwedge A$.

Chapter 2: Ordered Vector Spaces

Definition 28 - A *field* is a set $\mathbb{F}$ with elements $0,1$ and operations $(.)+(.): V \times V \to V$, $(.)\cdot(.): V \times V \to V$, $(.)^{-1}: V \setminus \{0\} \to V$ such that:

Closure

1) $\forall a,b \in \mathbb{F}, a+b \in \mathbb{F}$ (addition operation must be closed).

2) $\forall a,b \in \mathbb{F}, a \cdot b \in \mathbb{F}$ (multiplication operation must be closed).

Additive axioms

1) $a+(b+c)=(a+b)+c$ (associativity)

2) $a+b=b+a$ (commutativity of addition)

3) $a+0=a$ (zero element)

4) $a+(-a)=0$ (additive inverse)

5) $a \cdot \left(b+c\right)=a \cdot b + a \cdot c$ (distributivity)

Multiplicative axioms

1) $a \cdot (b \cdot c)=(a \cdot b) \cdot c$ (associativity of multiplication)

2) $a \cdot b = b \cdot a$ (commutativity of multiplication)

3) $a \cdot 1 = a$ (unit element)

4) $a \cdot (a^{-1}) = 1, a \neq 0$ (multiplicative inverse)

5) $(a+b) \cdot c = a \cdot b + a \cdot c$ (distributivity)

Definition 29 - A *vector space over a field* $\mathbb{F}$ is a set V with two operations $(.)+(.): V \times V \to V$, $(.)\circ(.): \mathbb{F} \times V \to V$ such that the following axioms are satisfied.

Additive axioms (Abelian group)

1) $\forall x, y \in V, x + y = y + x$ (commutativity)

2) $\forall x, y, z \in V, \left(x + y\right) + z = x + \left(y + z\right)$ (associativity)

3) $\forall x \in V, x + 0 = x$ (existence of additive zero)

4) $\forall x \in V, \exists \left(-x\right) \in V, x + \left(-x\right) = 0$ (existence of additive inverse)

Multiplicative axioms

5) $\forall x \in V, 1 \circ x = x$

6) $\forall x \in V, \forall c, d \in \mathbb{F}, \left(c \cdot d\right) \circ x = c \circ \left(d \circ x\right)$

Distributive axioms

7) $\forall x, y \in V, \forall c \in \mathbb{F}, c \circ \left(x + y\right) = c \circ x + c \circ y$

8) $\forall x \in V, \forall c, d \in \mathbb{F}, \left(c + d\right) \circ x = c \circ x + d \circ x$

Closure

9) $\forall x, y \in V, x + y \in V$

10) $\forall x \in V, \forall c \in \mathbb{F}, c \circ x \in V$

Definition 30 - A *real vector space* is a vector space $\left(V, \mathbb{F}, \circ, +\right)$ over the field $\mathbb{R}$.

Definition 31 - An *ordered vector space* V is a real vector space equipped with a partial order $\leq$ such that for all $x, y, z \in V$ and $0 \leq \lambda$ in $\mathbb{R}$ the following two axioms are satisfied

1) $x \leq y$ implies $x + z \leq y + z$ (translation invariance),

2) $x \leq y$ implies $\lambda \cdot x \leq \lambda \cdot y$ (positive homogeneity).

Definition 32 - ([7] Definition 1.14) **-** A *Riesz space* or vector lattice E is defined to be an ordered vector space which is a lattice i.e. for any pair of vectors $x, y \in E$ there exists their supremum and infimum.

Definition 33 - Let us consider a vector space V. A set K is called a *positive cone* $K \subseteq V$ if it satisfies the following conditions:

1) $K + K \subseteq K$,

2) $\mathbb{R}^+ K \subseteq K$,

3) $K \cap \left(-K\right) = \left\{0\right\}$.

If $\left(V, \leq\right)$ is a vector lattice then its positive cone is defined as $V^+ = \left\{u \in V : u \geq 0\right\}$.

Definition 34 - The cone of a Riesz space is called a *lattice cone*.

We shall also say that a cone K of a vector space X is a lattice cone if X partially ordered by the cone K is a Riesz space. In other words, the cone L_+ of an ordered vector space L is a lattice cone if and only if L is a Riesz space.

Example 11 - A vector lattice $\mathbb{R}^n$ is the space $\mathbb{R}^n$ with the standard order (i.e. $x \leq y \Leftrightarrow \forall_{i \in \{1,\ldots,n\}} x_i \leq y_i$). This order makes $\mathbb{R}^n$ a vector lattice in which $x \vee y$ and $x \wedge y$ are defined componentwise $\left(x \vee y\right)_i = x_i \vee y_i = \max\left\{x_i, y_i\right\}$, $\left(x \wedge y\right)_i = x_i \wedge y_i = \min\left\{x_i, y_i\right\}$.

Definition 35 - Let us consider a vector lattice $\left(V, \leq\right)$. The element $\left|x\right| = x \vee \left(-x\right)$ is called the *modulus* of $x \in V$.

Definition 36 - Let us consider a vector lattice $\left(V, \leq\right)$. The element $x^+ = x \vee 0$ is called the *positive part* of $x \in V$. The element $x^- = \left(-x\right) \vee 0$ is called the negative part of x.

Theorem 2 - If V is a real vector space and $K \subseteq V$ is a proper convex cone in V, there exists exactly one partial order on V that makes V into an ordered vector space such $V^+ = K$. This partial order is given by $x \leq y$ if and only if $x - y$ is in K.

Therefore, there exists a one-to-one correspondence between the partial orders on a vector space V that are compatible with the vector space structure and the proper convex cones of V.

Definition 37 - An ordered vector space is said to be *directly ordered* if for every two vectors $u, v \in V$ there exist $p, q \in V$ such that $u, v \in \left[p, q\right]$. This condition is equivalent to saying that $V = K - K$, i.e. the positive cone K is generating. Every vector lattice is directly ordered.

Definition 38 - A direct sum of the real vector lattices $\left(V_1, \leq_1\right), \left(V_2, \leq_2\right)$ is defined as the following set $V_1 \oplus V_2 = \left\{\left(v_1, v_2\right) : v_1 \in V_1, v_2 \in V_2\right\}$ with the following operations $\left(a_1, a_2\right) + \left(b_1, b_2\right) = \left(a_1 + b_1, a_2 + b_2\right)$, $\alpha\left(a_1, a_2\right) = \left(\alpha a_1, \alpha a_2\right)$ where $\alpha \in \mathbb{R}$.

The partial order is defined coordinate-wise.

$\left(a_1, a_2\right) \leq \left(b_1, b_2\right) \Leftrightarrow \left(a_1 \leq b_1\right) \text{and} \left(a_2 \leq b_2\right)$.

Presented definition can be extended to any number of real vector lattices

$\oplus_{i=1}^{n} V_i = V_1 \oplus V_2 \oplus ... \oplus V_n$.

In particular it is possible to define an ordered real vector space $\mathbb{R}^n = \oplus_{i=1}^{n} \mathbb{R}$ where

$\left(a_1, a_2, ..., a_n\right) \leq \left(b_1, b_2, ..., b_n\right) \Leftrightarrow \left(a_1 \leq b_1\right) \text{and} \left(a_2 \leq b_2\right) \text{and} ... \text{and} \left(a_n \leq b_n\right)$

$\left(a_1, a_2, ..., a_n\right), \left(b_1, b_2, ..., b_n\right) \in \mathbb{R}^n$.

Definition 39 - A vector subspace X of a vector lattice V is called a *lattice-subspace* if X equipped with the ordering from V is a vector lattice on its own (i.e. if the least upper bound of any two elements from X exists in X and the greatest lower bound of the elements).

Definition 40 - A vector $v \in V$, where V is a finite dimensional vector space ($n = \dim\left(V\right)$), is a *strictly positive vector* if for $i \in \left\{1, ..., n\right\}$ $v_i \geq 0$ and exists at least one v_j such that $v_j > 0$.

Definition 41 - The *closed line segment* connecting v and w is the set $\left[v,w\right]=\left\{x\in V : x=\lambda v+\left(1-\lambda\right)w, \lambda\in\left[0,1\right]\right\}$.

Definition 42 - An *interior point* of $\left[v,w\right]$ is any point in the closed line segment except for the end points v and w.

Definition 43 - A *set* $C\subset V$ is *convex*, if for all $v,w\in C$, $\left[v,w\right]\subset C$.

Definition 44 - *Conic* (nonnegative) combination of $x_1, x_2\in\mathbb{R}^n$ is any point of the form $x=\theta_1 x_1+\theta_2 x_2\in\mathbb{R}^n$ with $\theta_1\geq 0, \theta_2\geq 0$.

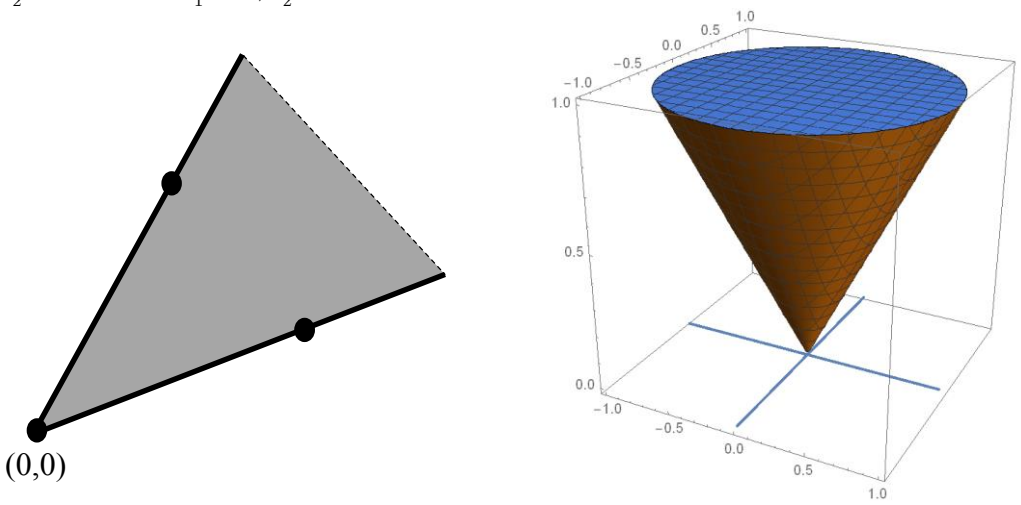

Figure 3: Simple examples of cones in $\mathbb{R}^2$ and $\mathbb{R}^3$.

Example 12 - Examples of cones:

1) $\left\{0\right\}$ - trivial cone,

2) $\text{ray}\left(v\right)=\left\{\lambda v : \lambda\geq 0\right\}$ - a ray generated by $v\neq 0$ which is the half line emanating form 0 through v.

3) Positive cone for the standard order (i.e., $x\geq y \Leftrightarrow \forall_{i\in\{1,2,\ldots,n\}} x_i\geq y_i$) is shown on the Figure 4.

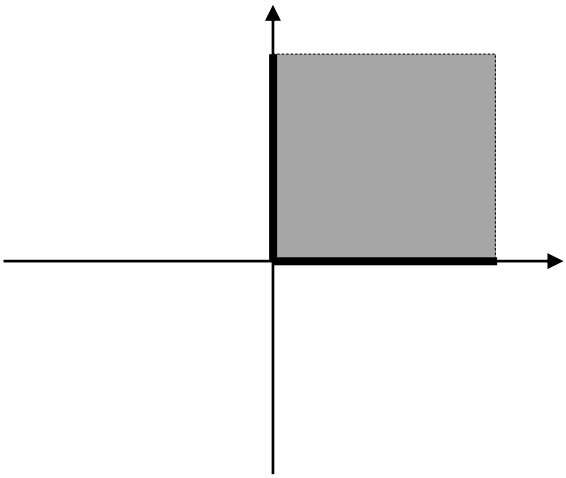

Figure 4: Positive cone for the standard order in $\mathbb{R}^2$.

Definition 45 - A *convex cone* K in a linear space V that is a cone that is also convex.

Theorem 3 - A set $K \subset V$ is a convex cone iff the following properties hold:

1) If $v \in K$ and $\lambda \geq 0$, then $\lambda v \in K$;

2) If $v, w \in K$, then $v + w \in K$.

Definition 46 - Let us consider a proper cone K, a *generalized inequality* can be defined in the following way:

$$x \geq y \Rightarrow x - y \in K \tag{11}$$

Example 13 - **Theorem 4** ([14] Theorem 2.1) - A cone in $\mathbb{R}^m$ is generating if and only if it has an interior point.

Definition 47 - *Orthogonal complement* $W^{\perp}$ of a subspace W of a vector space V equipped with an inner product $\cdot$ is the set $W^{\perp}$ of all vectors in V that are orthogonal to every vector in W. It is possible to write $W^{\perp} = \{v \in V : \forall w \in W, w \cdot v = 0\}$.

Theorem 5 ([14] Corollary 2.2) - For a subspace V of a finite dimensional vector space, exactly one of the following mutually exclusive possibilities holds:

1) V contains a strictly positive vector,
2) $V^{\perp}$ contains a nonnegative vector, where $V^{\perp}$ is the orthogonal complement of V.

Theorem 6 ([14] Corollary 2.3) - A subspace V of $\mathbb{R}^m$ has a generating cone if and only if $V^{\perp}$ contains no nonnegative vector.

Let's introduce a special the following notation for $x \in \mathbb{R}^m$. Let $x(i)$ denote the *i-th* component of x. Then the matrix whose rows are formed by the x_i can be written as

$$\begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1(1) & x_1(2) & \dots & x_1(m) \\ x_2(1) & x_2(2) & \dots & x_2(m) \\ \dots & \dots & \dots & \dots \\ x_n(1) & x_n(2) & \dots & x_n(m) \end{pmatrix}$$

From here we form the following m vectors of $\mathbb{R}^n$

$$y_1 = \begin{pmatrix} x_1(1) \\ x_2(1) \\ \dots \\ x_n(1) \end{pmatrix}, y_2 = \begin{pmatrix} x_1(2) \\ x_2(2) \\ \dots \\ x_n(2) \end{pmatrix}, \dots, y_m = \begin{pmatrix} x_1(m) \\ x_2(m) \\ \dots \\ x_n(m) \end{pmatrix}$$

Definition 48 ([14] Definition 2.4) - A set of n indices $\{m_1, \dots, m_n\}$ is called a *negative fundamental set* of indices for the vectors $x_1, \dots, x_n \in \mathbb{R}^m$ whenever

1) the n vectors $y_{m_1}, \dots, y_{m_n}$ are linearly independent; and
2) for at least one $j \in \{m_1, \dots, m_n\}$, all the coefficients in the expansion $y_j = \sum_{r=1}^{n} \alpha_{j,r} y_{m_r}$ are non-positive.

Definition 49 ([14]) - A solution ξ to the equation $b = \sum_{i=1}^{n} \xi_i x_i$ is called *basic nonnegative solution* if for the set $L = \{i : \xi_i > 0\}$, the set of vectors $\{y_i : i \in L\}$ is linearly independent.

Theorem 7 ([16] Theorem 2.11) - If the equation $b = \sum_{i=1}^{n} \xi_i x_i$ has a nonnegative solution, then it has a basic nonnegative solution.

Theorem 8 ([14] Lemma 2.6) - If $\alpha_1 y_1 + \dots + \alpha_m y_m = 0$, with $\alpha_i \geq 0$ and not all 0, and all $y_k \neq 0$, then there exists a subset $L \subseteq \{1, \dots, m\}$ such that there set S of vectors $\{y_i : i \in L\}$ is linearly independent and there is $j \notin L$ such that y_j is a negative linear combination of the vectors from S.

Theorem 9 ([14] Theorem 2.7) - The vector subspace V of $\mathbb{R}^m$ is directly ordered (has a generating cone) if and only if the vectors $x_1,...,x_n$ do not admit a negative fundamental set of indices $\{m_1,...,m_n\}$.

Definition 50 - Let K be a positive cone of an ordered space V. A vector $e \in K$ is called an *extremal vector* if $0 \leq x \leq e$ implied $x = \lambda e$ for some $\lambda \geq 0$.

Chapter 3: Exponential Time Method for Lattice Subspaces

Overview of The Method

In their paper [1], Abramovich, Aliprantis and Polyrakis gave necessary and sufficient conditions for a subspace of $\mathbb{R}^m$ to be lattice-ordered. We assume that the partial orders considered are coordinate-wise and that the subspace is $\langle X \rangle$, a subspace generated by a set X of n linearly independent positive vectors. We put the vectors from X in a $n \times m$ matrix as columns and consider the associated set $Y = \{y_1, y_2, ..., y_n\}$ of the rows of the matrix. Their main Theorem 2.6 asserts that $\langle X \rangle$ is lattice-ordered if and only if the set X admits a *fundamental set of indices* I, which means that the subset $Y_I \subseteq Y$ of vectors indexed by I is linearly independent, and every vector from $Y \setminus Y_I$ is a nonnegative linear combination of vectors from Y_I. The authors also give a computer algorithm that, based on the above result, determines if a given subspace is lattice ordered. The algorithm requires $\binom{m}{n}$ steps, which grows exponentially with m.

COMPUTATIONAL COMPLEXITY

Theorem 10 - Computational complexity of the method described in the paper [1] is exponential.

Proof

Part 1: $\binom{n}{k} < \left(\frac{ne}{k}\right)^k$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{\cancel{1 \cdot 2 \cdot \ldots \cdot (n-k)}\,(n-k+1) \cdot \ldots \cdot (n-1) \cdot n}{k!\,\cancel{1 \cdot 2 \cdot \ldots \cdot (n-k)}} = \frac{(n-k+1) \cdot \ldots \cdot (n-1) \cdot n}{k!} =$$

$$= \frac{(n-k+1) \cdot \ldots \cdot (n-1) \cdot n}{k!} \frac{n^k}{n^k} = \frac{\cancel{(n-k+1) \cdot \ldots \cdot (n-1) \cdot n}}{\cancel{n \cdot n \cdot \ldots \cdot n}} \frac{n^k}{k!} =$$

$$= \left(\frac{n-k+1}{n}\right) \cdot \ldots \cdot \left(\frac{n-1}{n}\right) \cdot \frac{n}{n} \frac{n^k}{k!} = \left(1 - \frac{k-1}{n}\right)\left(1 - \frac{k-2}{n}\right) \cdot \ldots \cdot \left(1 - \frac{1}{n}\right) \cdot 1 \frac{n^k}{k!}$$

By assumption $0 \le k \le n$ then $0 \le k \le n \Rightarrow 1 \ge 1 - \frac{k}{n} \ge 0$.

In particular for $k = 1$ we have $1 \ge 1 - \frac{1}{n} \ge 0$. For $k = k-1, 0 \le k-1 \le n$ we have $1 \ge 1 - \frac{k-1}{n} \ge 0$ and $1 \ge \left(1 - \frac{k-1}{n}\right)\left(1 - \frac{k-2}{n}\right) \cdot \ldots \cdot \left(1 - \frac{1}{n}\right)$ then

$$1 \ge \left(1 - \frac{k-1}{n}\right)\left(1 - \frac{k-2}{n}\right) \cdot \ldots \cdot \left(1 - \frac{1}{n}\right) \qquad /* \frac{n^k}{k!}$$

$$\frac{n^k}{k!} \ge \left(1 - \frac{k-1}{n}\right)\left(1 - \frac{k-2}{n}\right) \cdot \ldots \cdot \left(1 - \frac{1}{n}\right) \frac{n^k}{k!}$$

$$\left(1 - \frac{k-1}{n}\right) \cdot \ldots \cdot \left(1 - \frac{1}{n}\right) \cdot 1 \frac{n^k}{k!} \le \frac{n^k}{k!}$$

We know that $e^n = \sum_{k=0}^{\infty} \frac{n^k}{k!}$ and $e^k = \sum_{n=0}^{\infty} \frac{k^n}{n!}$ then

$$e^k = \sum_{n=0}^{\infty} \frac{k^n}{n!} = 1 + k + \frac{k^2}{2!} + \ldots + \frac{k^{k-1}}{\left(k-1\right)!} + \frac{k^k}{k!} + \frac{k^{k+1}}{\left(k+1\right)!} + \ldots > \frac{k^k}{k!}$$

$$e^k > \frac{k^k}{k!}$$

$$e^k \frac{k!}{k^k} > \frac{k^k}{k!}\frac{k!}{k^k}$$

$$e^k \frac{k!}{k^k} > 1$$

then $\left(\left(\left(\binom{n}{k} < \frac{n^k}{k!}1\right) \text{and} \left(1 < e^k \frac{k!}{k^k}\right)\right) \Rightarrow \binom{n}{k} \le \frac{n^k}{k!} \le \frac{n^k}{k!} \cdot 1 \le \frac{n^k}{k!} e^k \frac{k!}{k^k} = \frac{n^k e^k}{k^k} = \left(\frac{ne}{k}\right)^k\right.$.

Finally we can write $\binom{n}{k} < \left(\frac{ne}{k}\right)^k$.

Part II, $\left(\frac{n}{k}\right)^k \le \binom{n}{k}$

$$\binom{n}{k} = \frac{n!}{k!\left(n-k\right)!} = \frac{\cancel{1 \cdot 2 \cdot \ldots \cdot \left(n-k\right)}\left(n-k+1\right) \cdot \ldots \cdot \left(n-1\right) \cdot n}{k!\,\cancel{1 \cdot 2 \cdot \ldots \cdot \left(n-k\right)}} = \frac{\left(n-k+1\right) \cdot \ldots \cdot \left(n-1\right) \cdot n}{k!} =$$

$$= \frac{\left(n-\left(k-1\right)\right) \cdot \ldots \cdot \left(n-1\right) \cdot n}{\left(k-1\right)!k} = \frac{\cancel{\left(n-\left(k-1\right)\right)}\left(n-\left(k-2\right)\right) \cdot \ldots \cdot \left(n-1\right) \cdot n}{1 \cdot 2 \cdot \ldots \cdot \left(k-1\right)k} =$$

$$= \frac{n\prod_{i=1}^{k-1}\left(n-i\right)}{\left(k-\left(k-1\right)\right) \cdot \left(k-\left(k-2\right)\right) \cdot \ldots \cdot \left(k-1\right)k} = \frac{n\prod_{i=1}^{k-1}\left(n-i\right)}{k\prod_{i=1}^{k-1}\left(k-i\right)} = \frac{n}{k}\prod_{i=1}^{k-1}\frac{n-i}{k-i}$$

Let assume that $k \le n$ then

$$k \le n \Rightarrow -k \ge -n \Rightarrow -ki \ge -ni \Rightarrow kn - ki \ge nk - ni \Rightarrow k\left(n-i\right) \ge n\left(k-i\right)$$

$$k\left(n-i\right) \ge n\left(k-i\right) \Rightarrow \frac{k\left(n-i\right)}{n\left(k-i\right)} \ge \frac{n\left(k-i\right)}{n\left(k-i\right)} \Rightarrow \frac{k\left(n-i\right)}{n\left(k-i\right)} \ge 1 \Rightarrow \frac{n-i}{k-i} \ge \frac{n}{k}$$

$$\begin{cases} \frac{n}{k}\prod_{i=1}^{k-1}\frac{n-i}{k-i} \\ \frac{n-i}{k-i} \ge \frac{n}{k} \end{cases} \Rightarrow \frac{n}{k}\prod_{i=1}^{k-1}\frac{n-i}{k-i} \ge \frac{n}{k}\prod_{i=1}^{k-1}\frac{n}{k} = \frac{n}{k}\left(\frac{n}{k}\right)^{k-1} = \left(\frac{n}{k}\right)^k$$

$$\binom{n}{k} = \frac{n}{k}\prod_{i=1}^{k-1}\frac{n-i}{k-i} \geq \left(\frac{n}{k}\right)^k$$

and there are the following bounds for the binomial coefficients $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$.

Now let's assume that $k = \frac{n}{2}$ then $2^{\frac{n}{2}} \leq \binom{n}{\frac{n}{2}} \leq \left(2e\right)^{\frac{n}{2}}$ and it is clear that the complexity of the method is exponential. □

COMPUTER IMPLEMENTATION

The function K is defined in the file “K.m” [20].

```
%%%%%%%%%%%%%%%%%%%%
function lattice=K(x);
if any(any(x<0)) ~=0,
        lattice=0;
        return;
end
[N,M]=size(x);
if rank(x) ~=N,
        lattice=0;
        return;
end
z=combntns(1:M,N);
zdim=length(z);
lattice=0;
[l,u]=lu(x);
for j=1:zdim,
        k2=z(j,:);
        l2=setdiff(1:M,k2);
        b2=u(:,k2) \ (l\x(:,l2));
        if (sum((b2)>=0)==N)>0,
                lattice=1;
                disp(sprintf('\n fundamental set'))
        k2
```

```
        KK(N,M,b2,k2,l2);
        break;
end;
end;
```

%%%%%%%%%%%%%%%%%%%%

The function KK is defined in the file 'KK.m'.

%%%%%%%%%%%%%%%%%%%%

```
function lattice=KK(n,m,b,f,L);
diafora=m-n;
g=-20.*ones(n,m);
f2=L;
monad=eye(n);
k=1;
for i=1:m,
   for j=1:n,
   if f(1,j)==i, g(:,i)=monad(:,k); k=k+1;
   end;
end;
end;
k=1;
for i=1:m,
   for j=1:diafora,
      if f2(1,j)==i, g(:,i)=b(:,k); k=k+1;
      end;
   end;
```

```
end;
disp(sprintf('\n positive basis'))
g
%%%%%%%%%%%%%%%%%%%%
```

EXAMPLE 1

Examples 1-4 was published in the paper [20].

Data for the calculations $X = \begin{bmatrix} 1 & 1 & 0 & 4 & 1 \\ 2 & 3 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix}$.

Fundamental set of indices $I = \begin{bmatrix} 0, 2, 3, 4, 0 \end{bmatrix}$.

Lattice YES.

EXAMPLE 2

Data for the calculations $X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$.

Fundamental set of indices $I = \begin{bmatrix} 1 & 3 & 4 & 5 & 7 & 8 & 10 \end{bmatrix}$.

Lattice YES.

EXAMPLE 3

Data for the calculations $X = \begin{bmatrix} 2 & 2 & 4 & 3 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 2 & 3 & 1 & 3 & 4 & 4 \\ 3 & 3 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 6 \end{bmatrix}$

Fundamental set of indices $I = \begin{bmatrix} 1 & 3 & 4 & 5 & 6 & 7 & 9 \end{bmatrix}$.

Lattice YES.

EXAMPLE 4

Data for the calculations

$$X = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 3 & 4 & 1 & 10 & 11 & 60 & 0 & 12 & 4 & 32 & 13 \\ 2 & 1 & 40 & 30 & 2 & 23 & 4 & 5 & 2 & 9 & 12 & 1 & 1 & 1 & 5 & 33 & 14 \\ 3 & 10 & 20 & 10 & 3 & 24 & 5 & 6 & 3 & 8 & 13 & 2 & 2 & 2 & 6 & 34 & 15 \\ 4 & 30 & 30 & 0 & 4 & 25 & 6 & 7 & 0 & 7 & 14 & 3 & 3 & 3 & 7 & 35 & 16 \\ 5 & 40 & 1 & 0 & 5 & 0 & 7 & 8 & 5 & 6 & 15 & 0 & 0 & 3 & 8 & 0 & 17 \\ 6 & 50 & 0 & 0 & 6 & 27 & 8 & 9 & 0 & 5 & 16 & 50 & 11 & 12 & 5 & 37 & 18 \\ 7 & 1 & 1 & 1 & 7 & 28 & 9 & 10 & 0 & 4 & 17 & 0 & 40 & 5 & 4 & 38 & 19 \\ 8 & 3 & 0 & 0 & 1 & 29 & 10 & 11 & 1 & 3 & 18 & 10 & 0 & 10 & 3 & 39 & 20 \\ 9 & 50 & 0 & 40 & 2 & 30 & 11 & 12 & 12 & 2 & 19 & 0 & 10 & 10 & 2 & 40 & 21 \\ 10 & 70 & 40 & 1 & 3 & 31 & 12 & 13 & 13 & 1 & 20 & 2 & 2 & 2 & 1 & 41 & 22 \end{bmatrix}$$

Lattice NO.

Chapter 4: Feasible Algorithm for Lattice Subspaces

DESCRIPTION OF THE METHOD

In this section an algorithm, which is of polynomial time will be presented. We limit our input to algebraic numbers. We begin with the vector INDEX that contains a subset **I** of indices and the function **PREFUND**(Y,INDEX) that return a subset Y_I of Y indexed by *I*, respectively. There are known (c.f. [11]) polynomial-time algorithms that solve linear programming problems. Therefore we can have a polynomial-time (boolean output) routine **NONNEGCOMB**($y[i], Z$) that checks if a vector $y[i] \in Y$ is a nonnegative linear combination of vectors from $Z \subset Y$ [13].

GET_FUNDAMENTAL_INDEX(m, Y)

{

INDEX:= $\{1,...,m\}$;

$Z := Y$;

for($i := 1$; $i <= m$; $i++$)

if **NONNEGCOMB**($y[i], Z$)

{ $Z := Z \setminus \{y_i\}$;

INDEX := *INDEX* $\setminus \{i\}$; }

return *INDEX*, **PREFUND**(Y ,*INDEX*);

}

Also, there are known polynomial-time algorithms checking linear independence of a set of vectors. Let **LININDEP**(Z) be such an algorithm checking linear independence of a set Z of vectors. Now our main algorithm **LATTICE**(m, X , *INDEX*) can be written. Let **GETY**(X) be an algorithm returning the associated set Y given the input X.

LATTICE(m, X , *INDEX*)

{

$Y :=$ **GETY**(X);

if **LININDEP**(**PREFUND**(Y ,*INDEX*))

IS_LATTICE := yes;

else

IS_LATTICE := no;

return *IS_LATTICE;*

}

COMPUTATIONAL COMPLEXITY OF THE LINEAR PROGRAMMING PROBLEM

Presented approach is based on the linear programing optimization method. Linear programs are problems that can be expressed in the following canonical form:

$$\min c^T x \quad s.t. \begin{cases} Ax \le b \\ x \ge 0 \end{cases} \tag{12}$$

where x represents the vector of variable (to be determined), c and b are known vectors of coefficients, A is a known matrix of coefficients [18].

The simplex algorithm was developed by George Dantzig in 1947, solves LP problems by constructing a feasible solution at a vertex of the polytope and then walking along a path on the edges of the polytope to vertices with non-decreasing values of the objective function until an optimum is reached for sure [12]. The simplex algorithm has poor worst-case behavior: Klee and Minty constructed a family of linear programming problems for which the simplex method takes a number of steps exponential in the problem size [26]. According to numerical experiments, the simplex algorithm is quite efficient. The simplex algorithm has been proved to solve "random" problems efficiently, i.e. in a cubic number of steps [9].

The linear programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979 [21], but a larger theoretical and practical breakthrough in the field came in 1984 when Narendra Karmarkar introduced a new interior-point method for solving linear programming problems [19]. Karmarkar presented an $O\left(nL\right)$ (L is the number of the bits in the input parameters) iterations algorithm with $O\left(n^3\right)$ work per iteration, thus resulting in an $O\left(n^4L\right)$ algorithm [8]. Interior-point methods move through the interior of the feasible region by using different strategies. Many interior-point methods have been proposed and analyzed. For both theoretical and practical purposes, barrier function or path-following methods have been the most popular since the 1990s.

In presented work the MATLAB implementation of the interior point method have been applied [24]. In the MATLAB environment the default method for the solution of the Linear Programming problem is the interior-point method based on LIPSOL [27], which is a variant of Mehrotra's predictor-corrector algorithm [25], a primal-dual interior-point method. The central trajectory can be followed to the optimal solution in $O\left(\sqrt{n}L\right)$ iterations. According to the numerical experiments performance of the MATLAB implementation of the procedure for the solution of the linear programming problem is comparable with a FORTRAN code [27].

COMPUTER IMPLEMENTATION

MATLAB implementation of the method which was presented in the chapter 3 is given below.

```
%*****************************************
X=dlmread('X.txt');
                        %Read the n x m matrix X from the file 'X.txt'
                        %Each vector x(i) is in a separate row
n=size(X,1);            % n is a number of rows in the matrix X
m=size(X,2);            % m is a number of columns in the matrix X
indexSize=m-1;          % indexSize is a number of the vectors Y[i] in the collection Y_I
v=zeros(m,1);
b=zeros(n,1);

for i=1:m
   Z=zeros(n, indexSize);
   lb=zeros(indexSize,1);
   f=zeros(indexSize,1)+1;

   for j=1:n
      b(j)=X(j,i);
   end

   q=0;
   for k=1:m
```

```
        if (( v(k)==0 )&&(k ~= i))
            q=q+1;
            for j=1:n
                Z(j,q)=X(j,k);
            end
        end
    end

    % linear programming
    opts = optimset('display','off');
    [x,fval,exitflag,output,lambda] = linprog(f,[],[],Z,b,lb,[],[],opts);

    if( exitflag == 1 )
       v(i)=1;
       indexSize = indexSize-1;
    end
end

indexSize = indexSize +1;
Z=zeros(n, indexSize);
for i=1:m
    q=0;
    for k=1:m
        if ( v(k)==0 )
            q=q+1;
```

```
            for j=1:n
                Z(j,q)=X(j,k);
            end
        end
    end
end

r=rank(Z);
if( r == m1 )
    INDEX1=zeros(indexSize,1);
    if(m-m1>0)
        INDEX2=zeros(m- indexSize,1);
    end
    disp('Lattice YES');
    counter1 = 0;
    counter2 = 0;
    for i=1:m
        if(v(i)==0)
            v(i)=i;
            counter1=counter1+1;
            INDEX1(counter1)=i;
        else
            v(i)=0;
            counter2=counter2+1;
            INDEX2(counter2)=i;
```

```
        end
    end

    dlmwrite('INDEX1.txt',INDEX1);
        if(m-m1>0)
            dlmwrite('INDEX2.txt',INDEX2);
        end
    dlmwrite('INDEX3.txt',v);
    fileID = fopen('lattice-YES','w');
    fclose(fileID);
else
    disp('Lattice NO');
    fileID = fopen('lattice-NO','w');
    fclose(fileID);
end
%*****************************************
```

EXAMPLE 1

Data for the calculations

$$X = \begin{bmatrix} 1 & 1 & 0 & 4 & 1 \\ 2 & 3 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Fundamental set of indices $I = \begin{bmatrix} 0,2,3,4,0 \end{bmatrix}$, $Y_I = \begin{bmatrix} 1 & 0 & 4 \\ 3 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$.

Lattice YES.

EXAMPLE 2

Data for the calculations

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Fundamental set of indices

$$I = \begin{bmatrix} 1 & 3 & 4 & 7 & 8 & 9 & 10 \end{bmatrix}, \; Y_I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Lattice YES.

EXAMPLE 3

Data for the calculations

$$X = \begin{bmatrix} 2 & 2 & 4 & 3 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 2 & 3 & 1 & 3 & 4 & 4 \\ 3 & 3 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 6 \end{bmatrix}.$$

Fundamental set of indices

$$I = \begin{bmatrix} 2 & 3 & 4 & 5 & 7 & 8 & 10 \end{bmatrix}, \; Y_I = \begin{bmatrix} 2 & 4 & 3 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 2 & 1 & 3 & 4 \\ 3 & 0 & 0 & 0 & 4 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 \end{bmatrix}.$$

Lattice YES.

EXAMPLE 4

Data for the calculations

$$X = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 3 & 4 & 1 & 10 & 11 & 60 & 0 & 12 & 4 & 32 & 13 \\ 2 & 1 & 40 & 30 & 2 & 23 & 4 & 5 & 2 & 9 & 12 & 1 & 1 & 1 & 5 & 33 & 14 \\ 3 & 10 & 20 & 10 & 3 & 24 & 5 & 6 & 3 & 8 & 13 & 2 & 2 & 2 & 6 & 34 & 15 \\ 4 & 30 & 30 & 0 & 4 & 25 & 6 & 7 & 0 & 7 & 14 & 3 & 3 & 3 & 7 & 35 & 16 \\ 5 & 40 & 1 & 0 & 5 & 0 & 7 & 8 & 5 & 6 & 15 & 0 & 0 & 3 & 8 & 0 & 17 \\ 6 & 50 & 0 & 0 & 6 & 27 & 8 & 9 & 0 & 5 & 16 & 50 & 11 & 12 & 5 & 37 & 18 \\ 7 & 1 & 1 & 1 & 7 & 28 & 9 & 10 & 0 & 4 & 17 & 0 & 40 & 5 & 4 & 38 & 19 \\ 8 & 3 & 0 & 0 & 1 & 29 & 10 & 11 & 1 & 3 & 18 & 10 & 0 & 10 & 3 & 39 & 20 \\ 9 & 50 & 0 & 40 & 2 & 30 & 11 & 12 & 12 & 2 & 19 & 0 & 10 & 10 & 2 & 40 & 21 \\ 10 & 70 & 40 & 1 & 3 & 31 & 12 & 13 & 13 & 1 & 20 & 2 & 2 & 2 & 1 & 41 & 22 \end{bmatrix}.$$

Lattice NO.

EXAMPLE 5

Let's consider a family of $n \times m$ problems with the following X matrices where $m = 2n$. For $i = 1,...,n$ vectors y_i are defined in the following way

$$\left(y_i\right)_j = \begin{cases} i, & i = j \\ 0, & i \neq j \end{cases}, i = 1,...,n, \quad y_{n+i} = \left(i+1\right)y_1 + \left(i+1\right)y_2 + ... + \left(i+1\right)y_n, i = 1,...,n. \quad (13)$$

Sample matrix X for $n = 10, m = 20$ is shown below

$$X_1 = \left[y_1, y_2, ..., y_m\right] = \left[y_1, y_2, ..., y_{2n-1}, y_{2n}\right] \quad (14)$$

$$X_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 6 & 10 & 15 & 21 & 28 & 36 & 45 & 55 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 10 & 18 & 28 & 40 & 54 & 70 & 88 & 108 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 9 & 9 & 21 & 36 & 54 & 75 & 99 & 126 & 156 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 12 & 24 & 16 & 36 & 60 & 88 & 120 & 156 & 196 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 5 & 15 & 30 & 50 & 25 & 55 & 90 & 130 & 175 & 225 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 6 & 18 & 36 & 60 & 90 & 36 & 78 & 126 & 180 & 240 \\ 0 & 0 & 0 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 7 & 21 & 42 & 70 & 105 & 147 & 49 & 105 & 168 & 238 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 8 & 24 & 48 & 80 & 120 & 168 & 224 & 64 & 136 & 216 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & 0 & 9 & 27 & 54 & 90 & 135 & 189 & 252 & 324 & 81 & 171 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 10 & 30 & 60 & 100 & 150 & 210 & 280 & 360 & 450 & 100 \end{bmatrix}.$$

It is also possible to rearrange vectors $y_1, y_2, ..., y_m$ and form a new matrix X

$$X_2 = \left[y_1, y_{n+1}, y_2, y_{n+2}, ..., y_{n-1}, y_{n+n-1}, y_n, y_{n+n}\right]. \quad (15)$$

Sample result is shown below

$$X_2 = \begin{bmatrix}
1 & 1 & 0 & 3 & 0 & 6 & 0 & 10 & 0 & 15 & 0 & 21 & 0 & 28 & 0 & 36 & 0 & 45 & 0 & 55 \\
0 & 2 & 2 & 4 & 0 & 10 & 0 & 18 & 0 & 28 & 0 & 40 & 0 & 54 & 0 & 70 & 0 & 88 & 0 & 108 \\
0 & 3 & 0 & 9 & 3 & 9 & 0 & 21 & 0 & 36 & 0 & 54 & 0 & 75 & 0 & 99 & 0 & 126 & 0 & 156 \\
0 & 4 & 0 & 12 & 0 & 24 & 4 & 16 & 0 & 36 & 0 & 60 & 0 & 88 & 0 & 120 & 0 & 156 & 0 & 196 \\
0 & 5 & 0 & 15 & 0 & 30 & 0 & 50 & 5 & 25 & 0 & 55 & 0 & 90 & 0 & 130 & 0 & 175 & 0 & 225 \\
0 & 6 & 0 & 18 & 0 & 36 & 0 & 60 & 0 & 90 & 6 & 36 & 0 & 78 & 0 & 126 & 0 & 180 & 0 & 240 \\
0 & 7 & 0 & 21 & 0 & 42 & 0 & 70 & 0 & 105 & 0 & 147 & 7 & 49 & 0 & 105 & 0 & 168 & 0 & 238 \\
0 & 8 & 0 & 24 & 0 & 48 & 0 & 80 & 0 & 120 & 0 & 168 & 0 & 224 & 8 & 64 & 0 & 136 & 0 & 216 \\
0 & 9 & 0 & 27 & 0 & 54 & 0 & 90 & 0 & 135 & 0 & 189 & 0 & 252 & 0 & 324 & 9 & 81 & 0 & 171 \\
0 & 10 & 0 & 30 & 0 & 60 & 0 & 100 & 0 & 150 & 0 & 210 & 0 & 280 & 0 & 360 & 0 & 450 & 10 & 100
\end{bmatrix}.$$

Time of the calculations for a different value of *n* is shown in the Table 1 and Fig. 5.

The method presented in the chapter 3 produce equivalent results. Calculations was done on Dell Precision 690 with two quad-core processors Intel Xeon X5365 and 16 GB memory and MATLAB Version 8.0.0.783 R2012b.

Table 1: Time of the Calculations for the Polynomial and Exponential Time Method.

	Time of the calculations [s]	
n	Polynomial Time Method	Exponential Time Method
5	0.09	0.15
6	0.1	0.19
7	0.12	0.52
8	0.13	3.9
9	0.16	11.9
10	0.17	22.5
11	0.22	176
12	0.23	980
15	0.31	
20	0.5	
50	1.7	
70	3.5	
100	10	
120	19	
150	43.2	
200	140.6	

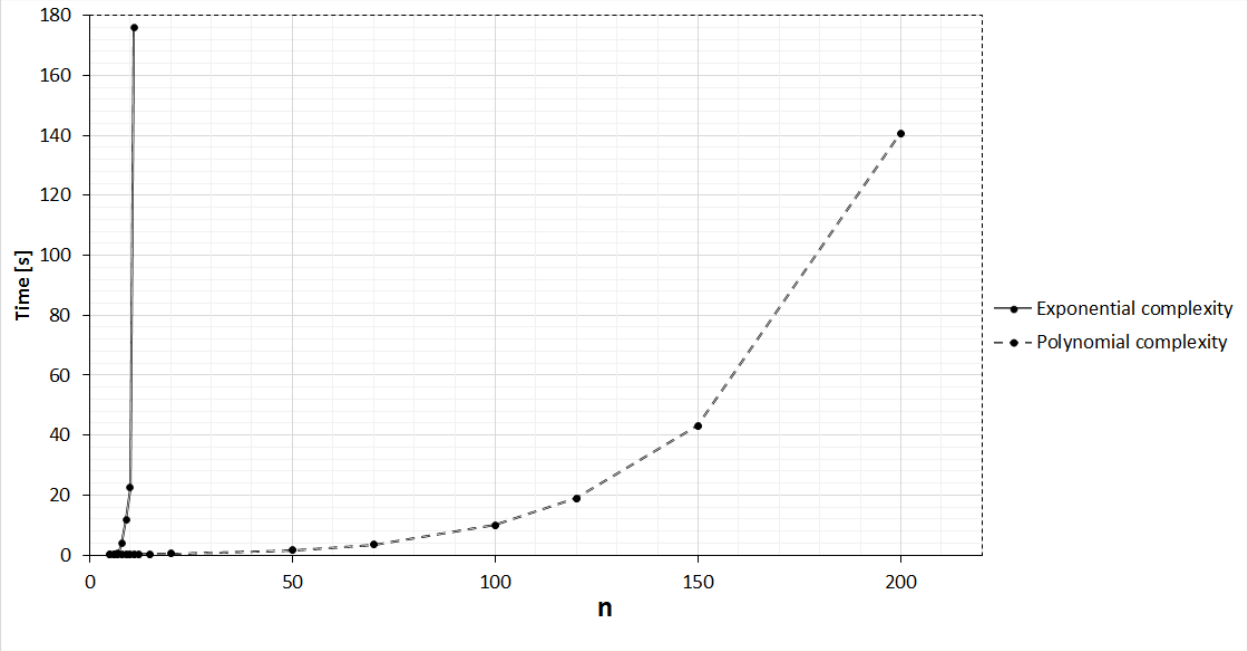

Figure 5: Time of the calculations for different values of n.

Chapter 5: Parallel Algorithm for Lattice Subspaces

DESCRIPTION OF THE METHOD

Parallel version of the method is based on the following theorem.

Theorem 11 - Let's assume that $y \in \mathbb{R}^n$ is not a positive linear combination of the vectors $\{y_1,...,y_m\}$ where $y_i \in \mathbb{R}^n$ then y is not a positive linear combination of any subset $\{y_{i_1},...,y_{i_k}\} \subset \{y_1,...,y_m\}$ where $k < n$ and $i_j \in \{1,...,m\}$ for $j = 1,2,...,k$.

Proof

Let assume that presented theorem is not true. In this case, y is not a positive linear combination of the vectors $y_1,...,y_m$ but it is a positive linear combination of the vectors $y_{i_1}, y_{i_2},..., y_{i_k}$ where $k < m$, and $i_p \neq i_q$ for $p \neq q$ i.e. $y = \alpha_{i_1} y_{i_1} + \alpha_{i_2} y_{i_2} + ... + \alpha_{i_k} y_{i_k} = \sum_{j=1}^{k} \alpha_{i_j} y_{i_j}$.
Now it is possible to introduce new indices

$$\beta_i = \begin{cases} \alpha_{i_j}, & \text{if } \exists_{i_j \in \{i_1,i_2,...,i_k\}} i_j = i \\ 0, & \text{if } \neg\left(\exists_{i_j \in \{i_1,i_2,...,i_k\}} i_j = i\right) \end{cases}, i \in \{1,2,...,n\} \tag{16}$$

and write $y = \sum_{i=1}^{n} \beta_i y_j$ i.e. y is a positive linear combination of the vectors y_i. This contradiction completes the proof of the theorem. □

A vector y_i is not a positive linear combination of the vectors $y_1, y_2,..., y_{i-1}, y_{i+1},..., y_m$ if no feasible solution of the following linear programming problem

$$\begin{cases} \min \sum_j \alpha_j \\ \begin{cases} \sum_{\substack{j=1 \\ j \neq i}}^{m} \alpha_j y_j = y_i \\ \alpha_j \geq 0 \end{cases} \end{cases} \tag{17}$$

exists. If y_i is not a positive linear combination of the vectors $y_1, y_2,..., y_{i-1}, y_{i+1},..., y_m$ then i is a member of the set of fundamental indices I and can be excluded from future calculations (according to the theorem 11). Linear programming problems (17) can be solved independently/simultaneously which can be used in parallel computing.

GET_FUNDAMENTAL_INDEX_PARALLEL(m, Y)

{

INDEX:= $\{1,\ldots,m\}_{1\times m}$;

MARK_ALL_VECTORS_FROM_INDEX_PARALLEL(*INDEX*, *Y*);

REMOVE_NONEGATIVE_COMBINATIONS_SERIAL(*INDEX*, *Y*);

return *INDEX,* **PREFUND**(*INDEX*, *Y*);

}

COMPUTER IMPLEMENTATION

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tic
X=dlmread('X.txt');

n=size(X,1);
m=size(X,2);

v=zeros(m,1);

m1=m-1;
parfor i=1:m
   pos = i;
   f=zeros(m1,1)+1;
   b=zeros(n,1);
   Z=zeros(n,m1);
   lb=zeros(m1,1);
   for j=1:n
       b(j)=X(j,pos);
   end
   q=0;
   for k=1:m
      if ( k ~= i )
         q=q+1;
         for j=1:n
```

```
            Z(j,q)=X(j,k);
        end
      end
   end
        opts=  optimset('display','off');
        [x,fval,exitflag,output,lambda] = linprog(f,[],[],Z,b,lb,[],[],opts);
   if( exitflag == -2 )
     v(i)=2;
   end
end

m1=m-1;
for i=1:m
        if(v(i)~=2)
                Z  = zeros(n,m1);
                lb = zeros(m1,1);
                f  = zeros(m1,1)+1;
                for j=1:n
                        b(j)=X(j,i);
                end
                q=0;
                for k=1:m
                        if ( (v(k)==0)||(v(k)==2) ) && (k~=i)
                                q=q+1;
                                for j=1:n
```

```
                    Z(j,q)=X(j,k);
                end
            end
        end
        opts=  optimset('display','off');
        [x,fval,exitflag,output,lambda] = linprog(f,[],[],Z,b,lb,[],[],opts);
        if( exitflag == 1 )
            v(i)=1;
            m1=m1-1;
        elseif( exitflag == -2 )
            v(i)=2;
        end
    end
end

m1   = 0;
for j=1:m
    if ( v(j) ~= 1 )
        m1   = m1 + 1;
    end
end
Z=zeros(n,m1);
q=0;
for k=1:m
   if ( v(k) ~= 1 )
```

```
        q=q+1;
        for j=1:n
            Z(j,q)=X(j,k);
        end
    end
end

r=rank(Z);

timeOfCalculations=toc;
fileID = fopen('time-p.txt','w');
fprintf(fileID,'%s',timeOfCalculations);
fclose(fileID);

if( r == m1 )
    INDEX1=zeros(m1,1);
    if(m-m1>0)
        INDEX2=zeros(m-m1,1);
    end
    disp('Lattice YES');
    INDEX3=zeros(m,1);

    counter1 = 0;
    counter2 = 0;
    for i=1:m
```

```
        if(v(i)~=1)
            INDEX3(i)=i;
            counter1=counter1+1;
            INDEX1(counter1)=i;
        else
            INDEX3(i)=0;
            counter2=counter2+1;
            INDEX2(counter2)=i;
        end
    end

    dlmwrite('INDEX1p.txt',INDEX1);
            if(m-m1>0)
                    dlmwrite('INDEX2p.txt',INDEX2);
            end
    dlmwrite('INDEX3p.txt',INDEX3);
    dlmwrite('INDEX4p.txt',v);
    fileID = fopen('lattice-YES-p','w');
    fclose(fileID);
else
    disp('Lattice NO');
    fileID = fopen('lattice-NO-p','w');
    fclose(fileID);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

EXAMPLE 1-2-3-4

The results for the examples 1,2,3,4 (chapter 3) are the same as in the chapter 3.

EXAMPLE 5

Let us consider a family of problems which is described in the example 5 from the chapter 3. Time of the calculations for serial and parallel method is shown in the Table 2.

Table 2: Time of the Calculations for the Serial and Parallel Method.

	Time of the calculations [s]	
n	Serial Method	Parallel Method
10	0.75	1.8
50	1.7	5.2
100	10	16.4
150	43.2	48.8
200	140.6	126
250	340.2	298
300	750	647
350	1449	1287

Calculations was done on Dell Precision 690 with two quad-core processors Intel Xeon X5365, 16 GB memory, MATLAB Version 8.0.0.783 R2012b, and 8 threads.

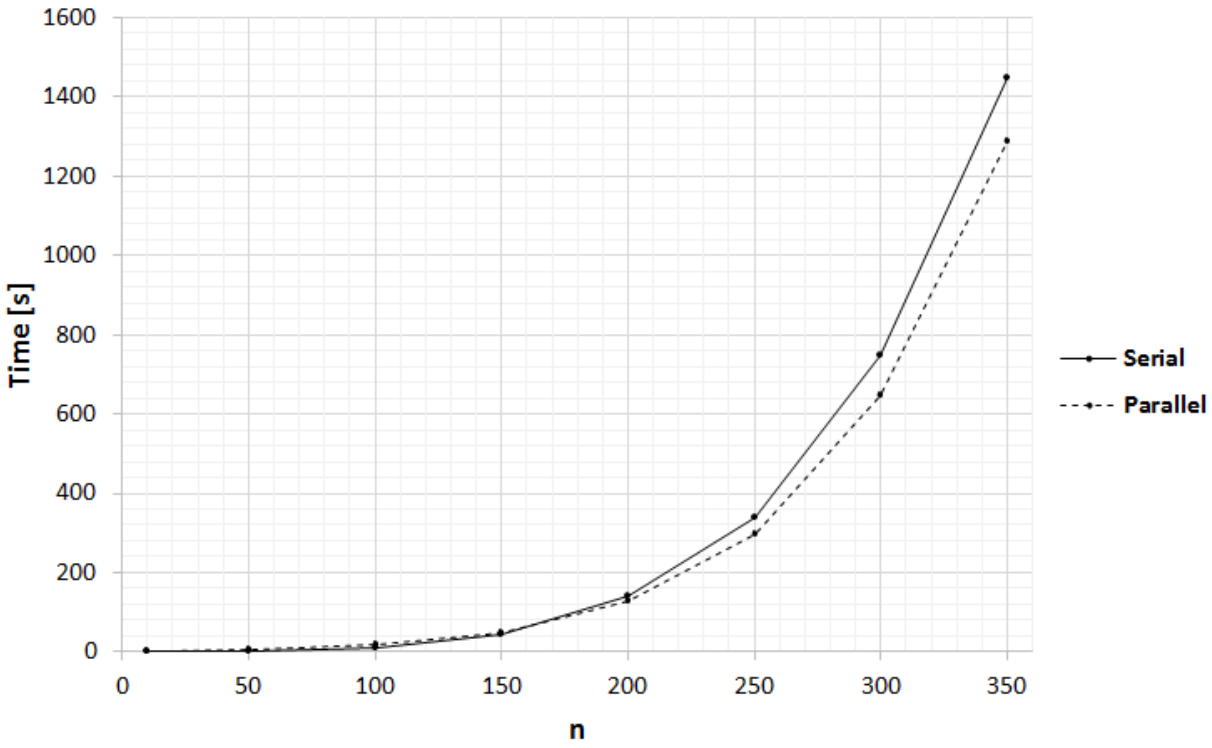

Figure 6: Time of the calculations for different values of n (8 threads).

EXAMPLE 6

Let's consider a family of $n \times (n+2)$ problems in the form

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 9 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 12 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 5 & 15 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 6 & 18 \\ 0 & 0 & 0 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 7 & 21 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 8 & 24 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & 0 & 9 & 27 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 10 & 30 \end{bmatrix}. \tag{18}$$

Columns y_i are defined in the equation (13). Comparison between the parallel and serial method is show in the Table 3 and the Figure 7.

Table 3: Time of the Calculations for the Serial and Parallel Method for 8 Threads.

	Time of the calculations [s]		
n	Serial Method	Parallel Method 4 Threads	Parallel Method 8 Threads
50	0.54	1.40	0.87
100	1.54	1.70	1.52
150	4.20	3.60	3.10
200	9.33	7.09	6.33
250	19.00	14.00	9.82
300	37.00	26.07	18.37
400	127.90	63.85	46.60
500	270.10	129.22	96.30

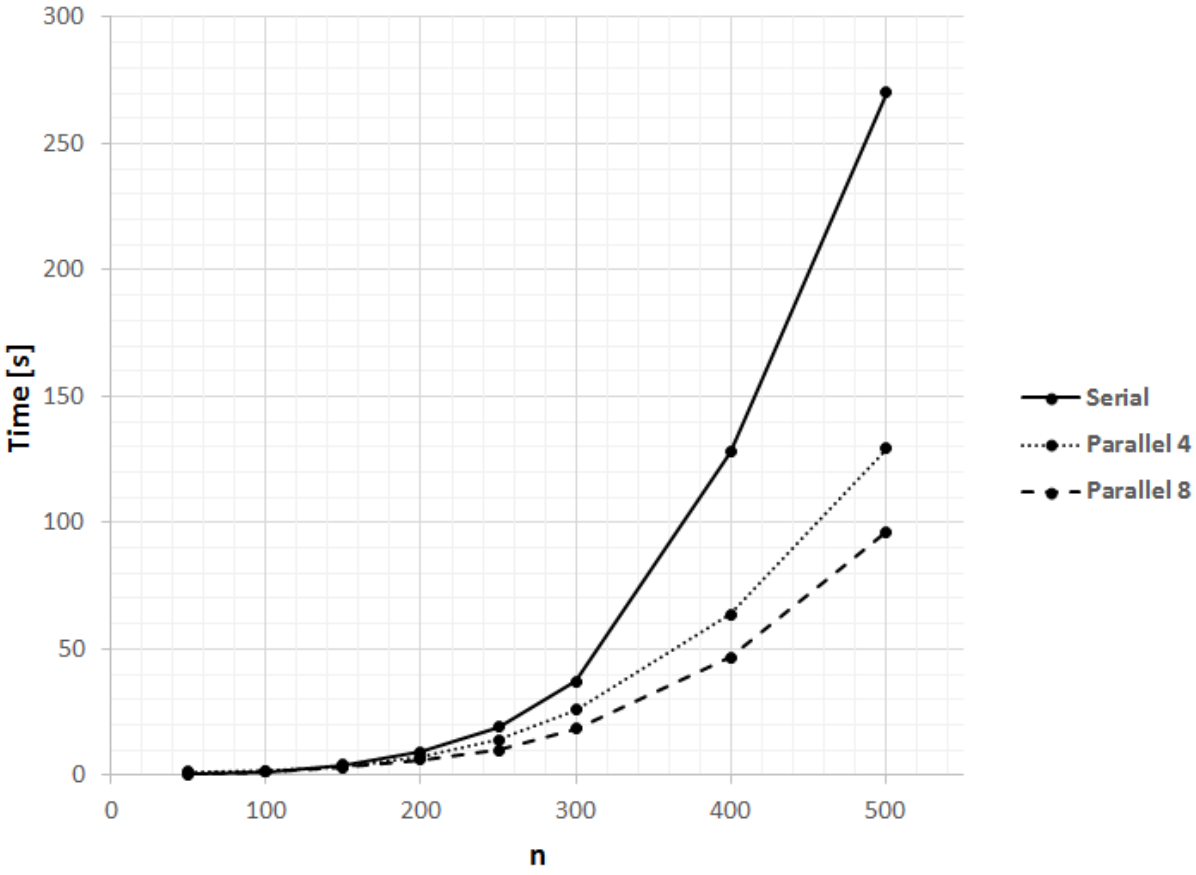

Figure 7: Time of the calculations for different values of n (4 threads and 8 threads).

Chapter 6: Applications

According to [20] lattice subspaces have been used in Mathematical Economics, in the analysis of incomplete markets and portfolio insurance [7][6]. There are also some application to the theory of security markets [22]. Other applications of the lattice subspaces in economics can be found in [2][4].

According to [5], portfolio insurance enables an investor to avoid losses while still capturing gains of portfolio payoff. It is possible to achieve by a proper application of the put options. It is possible to characterize the market structures in which the minimum-cost portfolio insurance is price independent by using the theory of Riesz spaces and lattice-subspaces [10]. It is proven that the minimum-cost insured portfolio exists if and only if the linear space generated by the corresponding financial instruments is lattice-ordered. A number of sufficient conditions for the asset span to be a lattice subspace is presented in [5]. The asset span of any two limited liability securities is a lattice subspace. Using a result form [1] Aliprantis et al. showed in [5] that an asset span is a lattice-subspace if and only if there exists a set of as many states as there are securities with the property that for any state not in that set the vector of payoffs of all securities is a linear positive combination of payoff vectors in states belonging to the set. The Aliprantis et al. [5] called such set a fundamental set of states.

Chapter 7: Conclusions

Known method for verification if a given subspace is a lattice ordered subspace of $\mathbb{R}^m$ can be applied for very small computational problems (n<20).

Serial method has polynomial complexity and can be effectively applied for large problems (n<500). In order to solve larger problems it is necessary to apply parallel computing.

In presented thesis theoretical background as well as numerical results were presented. Parallel method can be applied to the larger problems depending on available hardware. Current implementation of the parallel method is more effective than the serial method for sufficiently big *n*. More optimized parallel code written some HPC language (e.g. C/C++, FORTRAN) would be more effective.

It is possible to prove that the minimum-cost insured portfolio exists if and only if the linear space generated by the corresponding financial instruments is lattice-ordered and apply presented results in economics.

References

[1] Y.A. Abramovich, C.D. Aliprantis, and I.A. Polyrakis, Lattice-Subspaces and Positive Projections, Proceedings of the Royal Irish Academy, 94A, 2, 237-253, 1994.

[2] Y.A. Abramovich, C.D. Aliprantis, An Invitation to Operator Theory, American Mathematical Society, Providence, Rhode Island, 2002.

[3] I. Adler, N. Karmarkar, M.G.C. Resende, G. Veiga, An implementation of Karmarkar's algorithm for linear programming, Math. Programming, 44, 297-336, 1989.

[4] C.D. Aliprantis, K.C. Border, Infinite Dimensional Analysis, A Hitchiker's Guide, Springer-Verlag, Berlin, Heidelberg, New York, 1994.

[5] C.D. Aliprantis, D.J. Brown, and J. Werner, Minimum-cost portfolio insurance, Journal of Economic Dynamics & Control, Vol. 24, 1703-1719, 2000, DOI: 10.1016/S0165-1889(99)00091-3.

[6] C.D. Aliprantis, I.A. Polyrakis, R. Tourky, The cheapest hedge, J. Math. Econom. 37, 269–295, 2002.

[7] C. Aliprantis, R. C.-Tourky, Cones and Duality, in: Grad. Stud. Math., Vol. 84, Amer. Math. Soc., Providence, RI, 2007.

[8] D. Bertsimas, X. Luo, On the worst case complexity of potential reduction algorithms for linear programming, Mathematical Programming, 77, 321-333, 1997.

[9] K.H. Borgwardt, The Simplex Method: A Probabilistic Analysis (Algorithms and Combinatorics), Springer, 1987.

[10] D.J. Brown , S.A. Ross, Spanning, valuation and options. Economic Theory 1, 3-12, 1991.

[11] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, Introduction to Algorithms MIT, Press, Cambridge, MA, 2009.

[12] G.B Dantzig, Maximization of a linear function of variables subject to linear inequalities, 1947. Published pp. 339–347 in T.C. Koopmans (ed.): Activity Analysis of Production and Allocation, Wiley & Chapman-Hall, New York-London, 1951.

[13] J.J. Del Valle, V. Kreinovich, and P.J. Wojciechowski, Feasible algorithms for lattice and directed subspaces, Mathematical Proceedings of the Royal Irish Academy, 112A, 2, 199-204, 2014.

[14] J.J. Del Valle, P.J. Wojciechowski, A Note on Directly Ordered Subspaces of $\mathbb{R}^n$, Tatra Mt Math. Publ. 52, 101-113, 2012, 10.2478/v10127-012-0031-y.

[15] P.D. Domich, P.T. Boggs, J.R. Donaldson, and C. Witzgall, Optimal 3-dimensional methods for linear programming, NISTIR 89-4225, Center for Computing and Applied Mathematics, U.S. Dept. of Commerce, National Institute of Standards and Technology, Gaithersburg, MD, 1989.

[16] D. Gale, The Theory of Linear Economic Models. McGraw-Hill, New York, 1960.

[17] P.E. Gill, W. Murray, and M.A. Saunders, A single-phase dual barrier methodfor linear programming, Report SOL 88-10, Systems Optimization Laboratory, Stanford Univ., Stanford, CA, 1988.

[18] L.V. Kantorovich, A new method of solving some classes of extremal problems, Doklady Akad Sci USSR, 28, 1940, 211-214.

[19] N. Karmarkar, A new polynomial time algorithm for linear programming, Combinatorica, 4, 1984.

[20] V.N. Katsikis, Computational methods in portfolio insurance, Applied Mathematics and Computation, 189, 1, 9-22, 2007.

[21] L.G. Khachiyan. A Polynomial Algorithm in Linear Programming. Doklady Akademiia Nauk SSSR, 244, 1093–1096, 1979. (English translation: Soviet Mathematics Doklady, 20, 1, 191–194, 1979).

[22] C. Kountzakis, I.A. Polyrakis, The completion of security markets, Decisions Econom. Finance 29, 1–21, 2006.

[23] I.J. Lustigl, R.E. Marsten, and O.F. Shanno, Computational experience with a primal-dual interior point method for linear programming, TR J-89-11, Industrial and Systems Engineering Report Series, Georgia Inst. of Technology, Atlanta, GA, 1989.

[24] MathWorks, MATLAB, The Language of Technical Computing, http://www.mathworks.com.

[25] S. Mehrotra, On the Implementation of a Primal-Dual Interior Point Method, SIAM Journal on Optimization, 2, 575–601, 1992.

[26] Katta G. Murty, Linear programming. New York: John Wiley & Sons, Linear programming. New York: John Wiley & Sons, Inc., xix+482, 1983, ISBN 0-471-09725-X. MR 720547.

[27] Y. Zhang, Solving Large-Scale Linear Programs by Interior-Point Methods Under the MATLAB Environment, Department of Mathematics and Statistics, University of Maryland, Baltimore County, Baltimore, MD, Technical Report TR96-01, July, 1995.

Glossary

Vita

Andrew Martin Pownuk was born on August 19, 1969 in the town of Lubliniec, Poland. He entered the Silesian Technical University (Gliwice, Poland) in the Fall of 1990, and graduated five years later with a M.Sc./BS degree in Fundamental Technological Research with specialty in Applied Mechanics. In 1998 he completed his Postgraduate Study (specialty in computer networks, databases) from the Faculty of Automatic Control, Silesian University of Technology. In 2001 he received a Ph.D. degree from the Department of Civil Engineering, Silesian Technical University. Between 1995 and 2010 he worked for the Department of Theoretical Mechanics, Silesian University of Technology as a lecturer and an assistant professor. He have been working as a faculty at the Department of Mathematical Sciences at the University of Texas at El Paso since 2006.

Permanent address: 7244 Feather Hawk Dr.
El Paso, Texas 79912

This thesis/dissertation was typed by Andrew M. Pownuk.