An other Chubanov based algorithm for linear programming

Adrien Chan-Hon-Tong

An other Chubanov based algorithm for linear programming

Adrien CHAN-HON-TONG

November 19, 2019

Abstract

This paper presents an algorithm to solve linear programming based on Chubanov algorithm for homogeneous linear feasibility. This algorithm is different from the one described in arxiv.org/abs/1901.08525

1 Introduction

Linear programming is the very studied task of solving $\min_{x \ / \ Ax \geq b} c^T x$ for given $A \in \mathbb{Q}^{M \times N}$ a matrix, $b \in \mathbb{Q}^M$ and $c \in \mathbb{Q}^N$ some vectors.

Recently, [1] introduces an algorithm with good theoretical properties for homogeneous linear feasibility problems: for deciding if $\exists ? x \in \mathbb{Q}^N \ / \ Ax = \mathbf{0},\ x > \mathbf{0}$ given $A \in \mathbb{Q}^{M \times N}$ a full rank matrix. But, it is not known if one could take advantage of [1] for generic linear programming.

This paper explores this way by using [1].

Notations

$\mathbb{N}, \mathbb{Q}$ are the sets of integer and rational numbers. $\setminus$ is the ensemble subtraction. For all integers i, j, $[i, j]$ will symbolize the **integer** range i.e. $\{i, i+1, ..., j\}$ which is empty if $i > j$ (there will be no ambiguity with the interval in $\mathbb{R}$ as there is no real range in this paper).

For all integers i, j, I, J, $\mathbb{Q}^I$ is the set of I dimensional vectors on $\mathbb{Q}$, and, $\mathbb{Q}^{I \times J}$ is the set of matrix with I rows and J columns, with values in $\mathbb{Q}$, and, $._i$ designs the i component: a row for a matrix and a rational for vector or a row. $\mathbb{Q}^I$ would be matched with $\mathbb{Q}^{I \times 1}$ i.e. vectors are seen as columns, and, row of a matrix are matched with $\mathbb{Q}^{1,J}$. For all sets $S \subset \mathbb{N}$, A_S, b_S is the submatrix or subvector obtained when keeping only components indexed by $s \in S$. ^{T} is the transposition operation i.e. $A^T_{j,i} = A_{i,j}$. $\mathbf{0}$ and $\mathbf{1}$ are the 0 and 1 vector i.e. vector contains only 0 or only 1, and $\mathbf{I}$ is the identity matrix.

If $A \in \mathbb{Q}^{I \times J}$, the null vector space of A (i.e. the kernel) is written $Ker(A) = \{v \in \mathbb{Q}^J / Av = \mathbf{0}\}$, with the convention that Ker of empty A is all space.

Finally, it is written that complexity is $O(f(size))$ when there exists λ such that for all size $size$, worse case complexity of the algorithm on all instances of size $size$ is less than $\lambda f(size)$. The same stands for $\widetilde{O}(f(size))$ except that it means that there exists both λ and κ such that complexity is less than $\lambda f(size)\log(f(size))^{\kappa}$.

2 A new algorithm for linear programming

2.1 Key idea

The starting point of this algorithm is Chubanov algorithm [1] which solves efficiently $\exists ?x \in \mathbb{Q}^N \ / \ Ax > \mathbf{0}$ [2].

In this paper, the goal is to solve $Ax \geq b$. The idea is to build a vector v from each non optimal x such that moving along v leads to a point closer than a solution. Such vector v could be solved by computing a set I such that $A_I x$ can be improved. Indeed solving $A_I v > \mathbf{0}$ is possible using [2].

Of course, at first glance, it could take infinite times to reach the optimal solution by computing such v and updating $x = x + \varepsilon v$.

Yet, using the primal dual of the primal dual it seems that one could bound the number of such moves using prior on the problem.

2.2 Pre processing

Let recall the classical primal dual trick to go from optimization problem into a decision one.

The original goal of linear programming is to solve $\max_{A_{raw}x \leq b_{raw}, x \geq \mathbf{0}} c_{raw}^T x$. Then, it is well known that the dual problem is $\min_{A_{raw}^T y \geq c_{raw}, y \geq \mathbf{0}} b_{raw}^T y$. Now, the primal dual is formed by combining all constraints: $A_{raw}x \leq b_{raw}$, and, $x \geq \mathbf{0}$, and, $A_{raw}^T y \geq c_{raw}$, and $c_{raw}x = b_{raw}y$, and finally, $y \geq \mathbf{0}$. So, the problem $\max_{A_{raw}x \leq b_{raw}, x \geq \mathbf{0}} c_{raw}x$ can be folded into $\exists ?x \ / \ A_{big}x_{big} \geq b_{big}$ with

$$A_{big} = \begin{pmatrix} -A_{raw} & 0 \\ I & 0 \\ 0 & A_{raw}^T \\ 0 & I \\ c_{raw} & -b_{raw} \\ -c_{raw} & b_{raw} \end{pmatrix} \text{ and } b_{big} = \begin{pmatrix} -b_{raw} \\ 0 \\ c_{raw} \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

This primal dual problem has a solution i.f.f. the original polytope is neither empty or unbounded.

So the algorithm can assume without restricting the generality (as this pre processing is always possible) to want to solve $\exists ?x \ / \ Ax \geq b$.

Currently, this problem $\exists ?x \ / \ A_{big}x \geq b_{big}$ can be transformed by adding a variable z into $\min_{A_{big}x+z \geq b_{big}} z$. This last problem has the disadvantage of having twice the size of the original one, but, always admits admissible point (eventually $z = \max_m b_{big,m}$). Also, additional constraint $z \geq 0$ can or can not be added. When added, the

objective value becomes bounded. Now, the required pre processing of the offered algorithm is to take the primal dual $\exists ?x \ / \ A_{double}x \geq b_{double}$ of the first primal dual $\min_{A_{big}x+z\geq b_{big}, z\geq 0} z$ (i.e. doing to $\min_{A_{big}x+z\geq b_{big}, z\geq 0} z$ what was done on $\max_{A_{raw}x\leq b_{raw}, x\geq \mathbf{0}} c_{raw}^T x$). As the problem $\min_{A_{big}x+z\geq b_{big}, z\geq 0} z$ is neither empty or unbounded, then, it primal dual $\exists ?x \ / \ A_{double}x \geq b_{double}$ will always admit a solution x^* such that $A_{double}x^* \geq b_{double}$. So, if required the algorithm can even assume without restricting the generality (as this pre processing is always possible) to want to solve $\exists ?x \ / \ A_{double}x \geq b_{double}$ with the prior that such solution exists.

Algorithm 1 Massive Chubanov algorithm

input: $b \in \mathbb{Q}^M$, $A \in \mathbb{Q}^{M\times N}$
returns: returns x^* such that $Ax^* \geq b$, or, a certificate it is impossible
fails: on degenerated situations

while True **do**
  $F = \{m \in [1, M] \ / \ A_m x < b_m\}$
  if $F = \emptyset$ **then**
    return x
  Call Chubanov to find v such that $A_F v > \mathbf{0}$
  if there is no such v **then**
    return the certificate ($x^* - x$ would have worked)
  $\alpha = \max_{f\in F} \frac{b_f - A_f x}{A_f v}$
  while True **do**
    $\beta = \min_{m \ / \ A_m v<0} \frac{b_f - A_m x}{A_m v}$
    if $\beta > \alpha$ **then**
      $x = x + \frac{\alpha+\beta}{2} v$
      break
    if $\beta < \alpha$ **then**
      $E = \{m \in [1, M] \ / \ A_m v < 0 \text{ and } \frac{b_f - A_m x}{A_m v} = \beta\}$
      Call Chubanov to find w such that $A_{F\cup E} w > \mathbf{0}$
      if there is no such w **then**
        return the certificate ($x^* - (x + \frac{\alpha+\beta}{2} v)$ would have worked)
      $v = w$
      $\alpha = \max_{f\in F} \frac{b_f - A_f x}{A_f v}$
      continue
    if $\beta == \alpha$ **then**
      fails on this degenerated situation

2.3 Max min

In order to understand easily the algorithm, one can consider that our objective is to solve $\max_{x\in\mathbb{Q}^N} \min_{m\in[1,M]} A_m x - b_m$ with $A \in \mathbb{Q}^{M\times N}$ and $b \in \mathbb{Q}^N$.

Currently, our objective is just to exit when $\min_{m\in[1,M]} A_m x - b_m$ goes above 0. But, seen the problem as a max min can be useful.

Typically, in arxiv.org/abs/1901.08525, x is updated by looking for v such that $A_D v = \mathbf{1}$ with D the set of index reaching the min $\min_{m\in[1,M]} A_m x - b_m$ (sliding moves), and, when $A_D v = \mathbf{1}$ is not possible (simple projection), $A_D v > \mathbf{0}$ is computed instead using Chubanov algorithm.

This allows to have locally the strongest improvement of the min. However, drawback is that number of steps can be large.

Now, in this paper, the idea is to remark that it is possible to replace D by the set of **all** unsatisfied constraints $F = \{m \in [1, M] \ / \ A_m x < b_m\}$. Indeed, if the problem is feasible i.e. if exists x^* such that $Ax^* \geq b$, then, $A_F(x^* - x) > b_F - b_F > \mathbf{0}$.

So, if Chubanov returns a certificate of emptyness, then, one known that the original problem is either empty or unbounded. In other case, the algorithm can go on.

In addition, if $A_i x = b_i$, and that Chubanov returns that $A_{F\cup\{i\}} v > \mathbf{0}$ has no solution. Then, it means that $A_i x^* = b_i$ because otherwise $A_i x^* > b_i$, and, Chubanov should have returned something on $A_{F\cup\{i\}}$ (the inverse is not true but this is not a problem for our algorithm).

2.4 Step

So the central step is to compute $F = \{m \in [1, M] \ / \ A_m x < b_m\}$ and v such that $A_F v > \mathbf{0}$. Then, either moving along F makes a constraint exiting F (before constraints from $[1, M]\backslash F$ enters into F). Then, the step finishes successfully because F has decreased.

In the other case, let consider the first constraint i that becomes negative. First, if it is possible to have $A_{F\cup\{i\}} w > \mathbf{0}$. Then, one could proceed with w instead of v.

If not it means that there is no v such that $A_{F\cup\{i\}} v > \mathbf{0}$. But, there exists λ such that $A_{F\cup\{i\}}(x + \lambda v) < b_{F\cup\{i\}}$ as this constraint become negative before the reducing F. But, if the system was soluble than $A_{F\cup\{i\}}(x^* - (x + \lambda v)) > 0$ - this is a contradiction.

So, by moving along v

- either F strictly decreases
- either F strictly increase - but
  - either, it means the system has no solution because otherwise one would have $A_{F\cup\{i\}}(x^* - (x + \lambda v)) > 0$
  - either, i could be added to F

So, the only issue is when one constraint enters F exactly when one exits F. Yet, in this degenerated case, either one can be sure that one of these two constraints is saturated in optimal solution. This can be mitigated by disturbing v if the constraint that enters and the constraint that leaves are not equal (otherwise, this is an equality constraint).

2.5 Pseudo code

The pseudo code for linear program is presented in algorithm 1.

In reality only the first i can be pushed with F - because $F \cup \{i\}$ should have a solution, but $F \cup \{i, j\}$ may not. However, it seems possible to oscillate between i and j

3 Complexity

Algorithm offered in previous section does not solve linear program as it fails on degenerated situation (and as the loop 9-21 seems to be well defined only the first time).

However, when the algorithm terminates without failing, then, it terminates with a very good complexity:

- while loop from line 9 to 22 can only happens M times before returning into the main loop
- main loop forces one constraint from F to exit F (or fails on degenerated situation), so this loop can only happens M times
- so finally, complexity (to solve or fail) is $M^2\Upsilon$ where Υ is the complexity to compute χ such that $\Gamma\chi > \mathbf{0}$ (which has been showed very interesting thank to Chubanov algorithm [2])

Now, failing quickly is not very interesting... However, failure may be mitigated as condition line 21 can be discarded by adding noise in v (this is possible as $A_F v > \mathbf{0}$ not just $A_F v \geq \mathbf{0}$.

The only situation where adding noise will never work is if the constraint that enter is the opposite of the constraint that exits i.e. if there is equality constraints

So, this algorithm could become a real algorithm if updating the loop 9-21 for constraint which can push successively but non simultaneously and if equality constraints are handled carefully, and, if one could find a non stochastic way to perturbed v such that $\alpha == \beta$ is impossible (except for equality constraints). Currently, $v + \epsilon A_e$ for $e \in E$ seems a good way as $A_e A_i$ may be different than $A_e A_e$ for example if constraint are normalized. Yet, it is not clear if normalizing constraint while dealing with equality constraint is always possible...

However, this algorithm seems very interesting compared to other Chubanov based algorithm like arxiv.org/abs/1901.08525 which seems structurally forced to deal with many iterations...

References

[1] Sergei Chubanov. A polynomial projection algorithm for linear feasibility problems. *Mathematical Programming*, 153(2):687–713, 2015.

[2] Kees Roos. An improved version of chubanov's method for solving a homogeneous feasibility problem. *Optimization Methods and Software*, 33(1):26–44, 2018.