Could log barrier be implicit in barrier based linear programming solvers ?

Adrien Chan-Hon-Tong

Could log barrier be implicit in barrier based linear programming solvers ?

Adrien CHAN-HON-TONG

January 20, 2020

Abstract

Given a matrix $A \in \mathbb{Z}^{M \times N}$, computing an approximate solution of the unconstrained optimization problem $\min_{v \in \mathbb{R}^M} v^T A A^T v - \mathbf{1}^T \log(v)$ is a straightforward way to get a solution of the homogeneous linear program $Ax > \mathbf{0}$ (as general as linear programming). This can be done by newton method as the function is self concordant.

Yet, this method becomes very complex when dealing with matrix A with arbitrary large number: one has to carefully round the internal state at each step, but, such rounding is made difficult as evaluating the function is even not straightforward.

This paper wonder if minimizing directly $v^T A A^T v$ while maintaining $v > \mathbf{0}$ could be done in polynomial time. This could allow to remove (or precisely make implicit) the logarithm.

1 Introduction

Linear programming is the very studied task of optimizing a linear criterion under linear equality and inequality constraints. This problem has been first tackled by exponential algorithms like simplex [2] or perceptron [6]. Today, this problem is tackled by polynomial time algorithms like ellipsoid method [4, 3], log barrier method [5], or recently, Chubanov method [1].

1.1 Notation

The set of matrices of size $M \times N$ on $\mathbb{R}$ is written $\mathbb{R}^{M \times N}$ (the same for $\mathbb{Q}$ or $\mathbb{Z}$). This set is a vectorial space with an addition written $+$, and, a product between a scalar and a matrix written $\times$. If A is a matrix from $\mathbb{R}^{M \times N}$, then, the transposed matrix is written $A^T \in \mathbb{R}^{N \times M}$. The set of all matrices of any size has an inner product which is represented by juxtaposition of matrices: if $A \in \mathbb{R}^{I \times J}$, and, $B \in \mathbb{R}^{J \times K}$, then, AB is the matrix product of A and B which is in $\mathbb{R}^{I \times K}$. If A is a matrix, then A_i is the row i of A considered as a $1 \times N$ matrix. If I is the set of indexes, A_I is the submatrix when keeping only row i from i. Vectors of dimension N are considered as matrix with size $N \times 1$, but,

matrices 1×1 are considered as number: so, if v is a vector v_i is the value of component i. The product of a matrix A with a number l is written $l \times A$ but not lA as the number-matrix product is not a matrix-matrix product. $\mathbf{0}$, $\mathbf{1}$ are the vectors with all components being 0 or 1, and, $\mathbf{b}_i$ the i vector of the natural basis i.e. all components are 0 except component i with is 1. Finally, $Det(A)$ is the maximal determinant of any square submatrix of A (with maximal number of columns).

Greek letter are used to design abstract object (object which could not exist when proving contradiction, optimal solution which are not decidable, not rational numbers...).

1.2 Equivalence on linear programming

The native form of linear program is the task of solving $\min_{x \in \mathbb{Q}^N \ / \ Ax \geq b} cx$ for given $A \in \mathbb{Z}^{M \times N}$ a matrix, $b \in \mathbb{Z}^M$ and $c \in \mathbb{Z}^N$ some vectors.

But, it is well known that solving $\min_{x \in \mathbb{Q}^N \ / \ Ax \geq b} cx$ is equivalent to find a solution $y \in \mathbb{Q}^{M+N}$ (or prove there is not) to a system of inequality $Hy \geq h$ with ($H \in \mathbb{Z}^{(M+N+2) \times (N+M)}$ and $h \in \mathbb{Z}^{N+M+2}$) due to primal dual theory. There is a solution if and only if the original problem is both feasible and bounded. As finding (or prove there is not) y such that $Hy \geq h$ is equivalent to solve $\min_{z,t \in \mathbb{Q}^N \times \mathbb{Q} \ / \ Hz+t \geq h, t \geq 0} t$ which is a feasible and bounded problem, it is possible to consider primal dual again (on this last one). Thus, solver can assume that some solution exists, and, that the task is to find it. Then, as maximal determinant of a sub matrix is a polynomial in the binary size of the matrix, any solution of $Hz + t \geq h, t \geq 0, -t \geq -\varepsilon$ for a decidable ε can be converted into a solution by greedy projection. So, finding y (which exists) such that $Hy \geq h$ is equivalent to solve a system of strict inequality $Gz > g$. Finally, $Gz > g$ is equivalent to find x (not related to original problem) such that $\mathcal{A}x > \mathbf{0}$ because $\begin{pmatrix} G & -g \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ t \end{pmatrix} > \mathbf{0}$. This is also trivially equivalent to produce x such that $\mathcal{A}x \geq \mathbf{1}$.

This way, any linear programming solver can assume without restricting the generality to expect an input $A \in \mathbb{Z}^{M \times N}$, with the task of producing $x \in \mathbb{Q}^N$ such that $Ax > \mathbf{0}$, with the prior that such x exists.

Then, it is classical that one could remove extra variables until kernel of A ($Ker(A) = \{x \in \mathbb{Q}^N \ / \ Ax = \mathbf{0}\}$) is reduce to $\mathbf{0}$: if there exists x, χ such that $A\chi = \mathbf{0}$, $\chi \neq \mathbf{0}$ and $Ax > \mathbf{0}$, then, $A(x - \frac{x\chi}{\chi\chi}\chi) = Ax > \mathbf{0}$, and, $\chi(x - \frac{x\chi}{\chi\chi}\chi) = 0$. So, one could just find y such that $\mathcal{A}y > \mathbf{0}$ after injecting $y\chi = 0$ (i.e. $x_1 = -\frac{1}{\chi_1} \sum_{n=\{2,\dots,N\}} \chi_n x_n$ into $Ay > \mathbf{0}$ - this does not change the form of the problem).

Finally, if I, J is a partition of all indexes of rows of A, then, finding $A_J x > \mathbf{0}$, $A_I x = \mathbf{0}$, and, $A_I y > \mathbf{0}$ is sufficient to find z such that $Az > \mathbf{0}$. Indeed, even if $A_J y$ may not be acceptable positive, it will be possible to deal with it by adding

λx which does not modify $A_I y$. So if one find x, then it is only required to look for y (this strictly reduce the number of rows, and, possibly the number of variable as extra variable are removed). So, with all this reduction, one could consider reduced homogeneous linear feasibility only.

Definition: reduced homogeneous linear feasibility

Linear programming is equivalent to the task of finding $x \in \mathbb{Q}^N$ given $A \in \mathbb{Z}^{M \times N}$ such that $Ax \geq \mathbf{0}$, $x \neq \mathbf{0}$ given $A \in \mathbb{Q}^{M \times N}$ with $Ker(A) = \{\mathbf{0}\}$, and, with the prior that there exists $y \in \mathbb{R}^N$ such that $Ay > \mathbf{0}$.

2 Interior point

2.1 Mathematical description

This section describes the classical interior point method in context of homogeneous linear feasibility. Precisely, usually barrier function is used on the constraint $\log(A_i x - b_i)$ to forbid $A_i x \leq b_i$. This allows to to deal with problem in primal form but this requires to update cost function c during runtime. Here barrier function is only used on decomposition of x allowing to use a static problem, and, with unrelated constraints. This comes with the cost of casting the problem in homogeneous form. But this transformation is polynomial. So this is not an issue (it may in practice, as, homogeneous may have quadratic number of variables, constraints, and binary size - but it is ok in theory).

Let consider $F(v)$ from $]0, \infty[^M$ to $\mathbb{R}$, and, $x(v)$ from $]0, \infty[^M$ to $\mathbb{R}^N$ defined by:

$$x(v) = A^T v = \sum_{m=1}^{M} v_m A_m^T$$

$$F(v) = x(v)^T x(v) - \mathbf{1}^T \log(v)$$

$$= v^T A A^T v - \mathbf{1}^T \log(v) = \sum_{i,j=1}^{M} v_i v_j \times A_i A_j^T - \sum_{i=1}^{M} \log(v_i)$$

Let write $\frac{1}{v}$ for the vector in $\mathbb{R}^M$ such that value of component m is $\frac{1}{v_m}$, then, $\nabla_v F = Ax(v) - \frac{1}{v}$. The link with linear feasibility is that: if there exists $\nu \in]0, \infty[^M$ such that $\nabla_\nu F = \mathbf{0}$, then, $Ax(\nu) = \frac{1}{\nu} > \mathbf{0}$.

Even more, the two assumptions $||\nabla_v F|| \leq \delta$ and $\exists k$ such that $A_k x(v) < 0$ implies that $|A_k x(v)| + \frac{1}{v_k} \leq \delta$ i.e. $v_k \geq \frac{1}{\delta}$.

But, at this point, even for very small δ, this could be possible. Currently it is possible if exists v such that $x(v)^T x(v) = v^T A A^T v = 0$. But, as y exists, then AA^T is semi definite positive. So, there exists $\lambda > 0$ with binary size polynomial in binary size of A such that $x(v)^T x(v) = v^T A A^T v \geq \lambda v^T v$ when $v \geq \mathbf{0}$.

Main consequence is that $F(v) \geq \lambda_- v^T v - \mathbf{1}^T \log(v)$.

So, let introduce the function $f(t)$ from $]0, \infty[$ to $\mathbb{R}$ defined by $f(t) = \lambda_- t^2 - \log(t)$. This function goes to ∞ when t goes to 0 or to ∞ (t^2 dominates $\log(t)$).

So, this function admits a minimum $f^* = f(\tau) = 1 + \frac{1}{2}\log(\lambda)$ which corresponds to $f'(\tau) = 0$ for $\tau = \frac{1}{\sqrt{\lambda}}$.

By combining the two previous statements, $F(v) \geq Mf^*$, so F admit a minimum $F^* = M + \frac{M}{2}\log(\lambda)$ corresponding to $||\nabla_\nu F|| = 0$, and importantly, $F^* = F(\nu)$ is bounded by a polynomial in the binary size of A.

Also, as $f(t)$ goes to ∞ when t goes to 0 or to ∞, then, $f(t) \leq \phi$ implies $t \in [\alpha(\phi), \beta(\phi)]$. Precisely, $\alpha(\phi) = \exp(-\phi)$, because, $-\log(t) \leq f(t) \leq \phi$. For $\beta(\phi)$, either it is τ, or, β is the inverse of $f(t)$ on $[\tau, \infty[$. Currently, one can lower bound $f(t)$ by $f(\frac{2}{\sqrt{\lambda}}) + (t - \frac{2}{\sqrt{\lambda}})f'(\frac{2}{\sqrt{\lambda}})$ (this is a very coarse lower bound but $\log(\beta)$) is polynomial in the binary size of A, so...). Now, let consider F, and, an index k. Then, $F(v) \geq \lambda_- v^T v - \mathbf{1}^T \log(v) \geq (M-1)f^* + \lambda_- v_k^2 - \log(v_k)$. So, $\lambda_- v_k^2 - \log(v_k) \leq F(v) - (M-1)f^*$. So, a fortiori $\alpha(F(v) - (M-1)f^*) \leq v_k \leq \beta(F(v) - (M-1)f^*)$ at any point and for all k.

So, it is not possible to have simultaneously the three statements $F(v) < F(v_0)$, $||\nabla_v F|| \leq \frac{1}{\beta(F(v_0)-(M-1)f^*)}$ and $\exists k$ such that $A_k x(v) < 0$, because first assumption implies that $v_k < \beta(F(v_0) - (M-1)f^*)$ but last two imply that $v_k \geq \beta(F(v_0) - (M-1)f^*)$. So, if one find v such that $F(v) < F(v_0)$, and, $||\nabla_v F|| \leq \frac{1}{\beta(F(v_0)-(M-1)f^*)}$, then, $Ax(v) > \mathbf{0}$.

Now, F is self concordant (a sum of -log and a quadratic term). So, damped newton method can be applied on F.

This descent has two phases. First phase is characterized by a high gradient $(\nabla_v F)^T \nabla_v^2 F(\nabla_v F) \geq \frac{1}{4}$ but $F(v\_after\_update) \leq F(v) - \frac{1}{4}$. So, this phase can not last more than $4(F(v_0) - F^*)$ steps.

Let consider $v_0 = (\frac{1}{\sqrt{A_1 A_1^T}}, ..., \frac{1}{\sqrt{A_M A_M^T}})$ then as $A_i A_j \leq ||A_i||||A_j||$, it holds that $F(v_0) \leq M^2 + 2\sum_{m=1}^{M} \log(A_m A_m)$ i.e. $F(v_0)$ is polynomially bounded in the binary size of A. So, the number of step with $(\nabla_v F)^T \nabla_v^2 F(\nabla_v F) \geq \frac{1}{4}$ is polynomial in the binary size of A (as $4(F(v_0) - F^*)$ is).

Then, the algorithm enters in the second phase where $(\nabla_v F)^T \nabla_v^2 F(\nabla_v F) \leq \frac{1}{4}$. But, then, for every steps after this point, there is a quadratic converge of gradient to $\mathbf{0}$: $(\nabla_{v\_after} F)^T \nabla_{v\_after}^2 F(\nabla_{v\_after} F) \leq 2\left((\nabla_v F)^T \nabla_v^2 F(\nabla_v F)\right)^2$. So, one can make $\nabla_{v\_step\_t} F)^T \nabla_{v\_step\_t}^2 F(\nabla_{v\_step\_t} F \leq \varepsilon$ with $t \leq O(\log\log(\frac{1}{\varepsilon}))$. But, $(\nabla_v F)^T \nabla_v^2 F(\nabla_v F) \geq \exp(-2F(v_0)) \nabla_v F^T \nabla_v F$ because the hessian is AA^T (semi definite positive) plus $\frac{1}{v^2}$ a definite positive matrix bounded by $\exp(-2F(v_0))$. So, newton descent can make $\nabla_v F^T \nabla_v F$ small, precisely one can have $\nabla_v F^T \nabla_v F \leq \frac{1}{\exp(-2F(v_0))}\left((\nabla_v F)^T \nabla_v^2 F(\nabla_v F)\right)^2 \leq \varepsilon$ after a number of steps lower than $\log\log(\frac{1}{\varepsilon \exp(-2F(v_0))})$. For $\varepsilon = \frac{\lambda}{\beta(F(v_0)-(M-1)f^*)}$, one has the solution of the problem in a number of steps polynomial in the binary size of A.

2.2 Implementation issue

Now, let stress, this is not done yet: applying naive damped newton method will make binary size of v becoming exponential (and require to compute the root of $1+(\nabla_v F)^T \nabla_v^2 F(\nabla_v F)$ for the damped step). So, naively, each step will require exponential binary operation in binary size of A. This is why, it is important to do the damped newton descent while keeping an acceptable binary size.

Let consider v, v', v'' 3 successive points during the newton descent. From v the descent needs K steps, $K-1$ from v' and $K-2$ from v''. If one can round v'' such that $F(round(v''))$ is lower than $F(v')$, then, the descent should required less than $K-1$ steps from $round(v'')$ i.e. if rounding ensure that $F(round(v''))$ is lower than $F(v')$, then, the algorithm will convergence is less than $2K$ steps (where K is the number of steps for the infinite precision convergence).

So, the idea is to force denominator of even-step v to be $2^{-\Gamma}$ with $2^{-\Gamma}$ sufficiently small such that the error (less than $\frac{M}{2^{-\Gamma}}$) between native and rounded vectors does not make F to go higher than the corresponding odd-step v (numerator can be arbitrary large (as $v_k \leq \beta(F(v_0) - (M-1)f^*)$, numerator can not be higher than $2^{\Gamma + ceil(\log(\beta(F(v_0)-(M-1)f^*)))}$).

Hopefully, v is upper and lower bounded, then, F is lipschitz. Here, F is even lipschitz with a coefficient γ polynomial in the binary size of A. So, it is sufficient that $2^{M\gamma-\Gamma}$ is lower than the smallest newton improvement. Let ω be the smallest improvement. It is sufficient that $\Gamma \geq M\gamma - \log(\omega)$. And, ω is $\frac{1}{4}$ for first phase, and, related to the gradient in the second phase (so higher than ε from which one has created a solution).

Now, there is another issue which is that checking if $F(v) \leq F(w) - \varepsilon$ is even not that easy because it requires a careful approximation of log, and inner step of the damped newton descent requires to compute approximate square root !

This issue when dealing with arbitrary large number may plague numerous algorithm. Typically, ellipsoid method has a square root step, and, requires to round solution. Of course, this can be mitigated.

Yet, if it was possible to solve linear program using elementary function only, it could be interesting. Currently, Chubanov algorithm seems to have this feature. But, it is not clear for an interior like algorithm.

So, instead of relying on F. One could wonder if it is not possible to directly minimize $x(v)^T x(v)$ under constraint $v \geq \mathbf{0}$, $v^T \mathbf{1} \geq 1$. Let stress that this is very different from minimizing $x(v)^T x(v)$ under constraint $Ax(v) \geq \mathbf{1}$: constraints are linked in $Ax(v) \geq \mathbf{1}$, but, independent in $v \geq \mathbf{0}$ - only $v^T v = 1$ links the variable. Currently, the resulting solution may have nothing to do with a support vector machine solution related to A.

3 claim

The task is to find $x \in \mathbb{Q}^N$ given $A \in \mathbb{Z}^{M \times N}$ such that $Ax \geq \mathbf{0}$, $x \neq \mathbf{0}$ given $A \in \mathbb{Q}^{M \times N}$ with $Ker(A) = \{\mathbf{0}\}$, and, with the prior that there exists $y \in \mathbb{R}^N$

such that $Ay > \mathbf{0}$. Let

$$\nu = \underset{v \geq \mathbf{0}, v^T\mathbf{1} \geq 1}{\arg\min} \; v^T AA^T v = \underset{v \geq \mathbf{0}, v^T\mathbf{1} \geq 1}{\arg\min} \; x(v)^T x(v) \tag{1}$$

Let stress $y^T x(\nu) > 0$, as $\nu \geq \mathbf{0}$, $\nu^T\mathbf{1} \geq 1$. So, $x(\nu) \neq \mathbf{0}$, so $\nu^T AA^T \nu > 0$, so there is $m \in \{1, ..., M\}$ such that $A_m A^T \nu > 0$.

Now, if $\exists i \in \{1, ..., M\}$ such that $A_i A^T \nu < 0$, then, one could consider $A^T\nu - \frac{A_i A^T \nu}{\sqrt{A_i^T A_i}} \times A_i^T$ corresponding to $\phi = \nu - \frac{A_i A^T \nu}{\sqrt{A_i^T A_i}} \times \mathbf{b}_i$. Let stress that $\phi \geq \mathbf{0}$ because $A_i A^T \nu < 0$, and, also $\phi^T \phi \geq 1$. So, by definition of ν, $x(\phi)^T x(\phi) \geq x(\nu)^T x(\nu)$. But, $x(\phi)^T x(\phi) = x(\nu)^T x(\nu) - \frac{(A_i A^T \nu)^2}{2} < x(\nu)^T x(\nu)$. This is a contradiction.

Let $v \in \mathbb{Q}^{M \times N}$ with $v > \mathbf{0}$ and $v^T v = 1$. Either $AA^T v \geq \mathbf{0}$, and, one has a solution. Or, one could consider:

$$w = \underset{u^T\mathbf{1}=0}{\arg\min} \; (v+u)^T AA^T (v+u) + v^T AA^T v \times u^T \begin{pmatrix} \frac{1}{v_1^2} & 0 & 0 \\ & ... & \\ 0 & 0 & \frac{1}{v_M^2} \end{pmatrix} u \tag{2}$$

First, this equation has a solution because 0 is admissible, and, it is pure linear algebra (no inequality or no quadratic function).

Then, one could consider $\phi = \varepsilon \times \mathbf{b}_i - \varepsilon \times \mathbf{b}_m$ with $A_i A^T v < 0$, and, $A_m A^T v > 0$, at first order, $x(v+\phi)^T x(v+\phi) = x(v)^T x(v) + \varepsilon A_i A^T v - \varepsilon A_m A^T v$. So $x(v+\phi)^T x(v+\phi) < x(v)^T x(v)$ (at first order). But, $v^T \phi = 0$. So, the equation (2) as a not trivial solution w with $x(v+w)^T x(v+w) < x(v)^T x(v)$.

Now, if $w_i + v_i \leq 0$, it means that $w_i^2 \geq v_i^2$, but, this last point implies that $w^T \begin{pmatrix} \frac{1}{v_1^2} & 0 & 0 \\ & ... & \\ 0 & 0 & \frac{1}{v_M^2} \end{pmatrix} w \geq 1$. But, this leads necessarily to increase the cost function of equation (2). So, the solution of equation (2) verifies that $v + w > \mathbf{0}$. So, one could consider the algorithm consisting to compute w and update $v = v + w$.

Let notice that $w^T\mathbf{1} = 0$ implies that $(v+w)^T\mathbf{1} > v^T\mathbf{1} = 1$, so one could never reach $\mathbf{0}$, and, assumption on v are always true: $v > \mathbf{0}$, $v^T\mathbf{1} = 1$). So, this algorithm is well defined, and, as $x(v)^T x(v)$ is strictly decreasing, the algorithm does not loop.

Currently, the algorithm is very very close than previous interior point version, but, here the log is implicit, not explicit. One could monitor and/or perform rounding operations much more easily with this version ! Now, the question is the number of steps to converge.

Currently, one candidate for u is $\omega = \lambda \times (\nu - v)$ (so w give an even better reduction) with:

$$\lambda = \frac{v^T AA^T (v - \nu)}{(v-\nu)^T AA^T (v-\nu) + v^T AA^T v \times \mathbf{1}^T (1 - \frac{\nu}{v})^2}$$

leading to $x(v+\omega)^T x(v+\omega) = v^T AA^T v - \frac{(v^T AA^T(v-\nu))^2}{(v-\nu)^T AA^T(v-\nu)+v^T AA^T v \times \mathbf{1}^T(1-\frac{\nu}{v})^2}$ (so $x(v+w)^T x(v+w)$ is lower).

Obviously, this upper bound can be useless if $\mathbf{1}^T(1-\frac{\nu}{v})^2$ is arbitrary high, especially from a random initialization of v. Yet, in *good case* where $(1-\frac{\nu}{v})^2 \leq \Gamma((u-\nu)^T(u-\nu))^\gamma$, then, the dynamic of $x(v+\omega)^T x(v+\omega)$ is somehow:

$$value_{step+1} = value_{step}(1 - \frac{1}{1+\Gamma((u-\nu)^T(u-\nu))^\gamma})$$

$$value_{step+1} = value_{step}(\frac{1}{1+\frac{1}{\Gamma((u-\nu)^T(u-\nu))^\gamma}})$$

Such dynamic corresponds to a quadratic convergence. **This paper does not prove such convergence in general case.** It only claims that in *good case*, the convergence of v toward ν may be fast.

Currently, this convergence should probably be helped by trying to balance v_i by hand (when $A_i A^T v < 0$ it is always possible to do $v_i - = A_i A^T v$ plus root free normalization) and/or by forbidding v to try to reach some $\phi \in Ker(AA^T)$.

Important remarks:

For the convergence, it is not clear that $Ax(v) \geq \mathbf{0}$, even if $x(v)^T x(v)$ is arbitrary close to the optimum. Yet, even in this case, if $Ax(v) \geq -\frac{\mu}{2Det(A)}\mathbf{1}$ one can produce a solution.

$\mu = \min_{v \geq \mathbf{0}, v^T v \geq 1} \max_i (A_i x(v))^2$ is not nul, So, if $Ax(v) \geq -\frac{\mu}{2Det(A)}\mathbf{1}$, then, $Ax(\frac{1}{\mu} \times v) \geq -2Det(A) \times \mathbf{1}$, but, there is i such that $A_i x(\frac{1}{\mu} \times v) \geq 1$. So, one can consider $\min_{x,t \ Ax + t \times (\mathbf{1} - \mathbf{b}_i) \geq \mathbf{b}_i, t \geq 0} t$, this auxiliary problem can be initialized by $x(v)$ and $\frac{1}{2Det(A)}$. Just using greedy projection, one may reach a vertex with $t \leq \frac{1}{2Det(A)}$. As cramer rule apply on vertex, it means that $t = 0$ (so x is a solution of $Ax \geq \mathbf{b}_i$, and, so a solution of the original problem (notice that it is not necessarily matched with some v).

Also, equation (2) allows to decrease any v_i to almost 0, but, not to increase v_i much than $2v_i$. Yet, this can be mitigate, because, if $A_i A^T v < -v_i$, it is possible to do simple perceptron like update $v_i - = A_i A^T v$ (which decreases $x(v)^T x(v)$ while increasing exponentially v_i by a factor 2 - so there could not have more than a polynomial number of such steps). So, equation (2) will only be used on v such that $A_i A^T v + v_i > 0$.

Also, it could also be interesting to check if the minimizing of $v^T AA^T v$ when v is orthogonal to the kernel of A (this is just linear algebra: $\min_{v, Bv = \mathbf{0}} v^T AA^T v$, with B an orthogonal basis of the kernel). It seems that this means is an eigen vector linked with the minimal not nul eigen value. Yet, being an eigen vector implies constraint on v contrary to just being orthogonal to the kernel of A.

Finally, let remark that:

$$\nu_1 = \arg\min_{v \geq \mathbf{0}, v^T \mathbf{1} \geq 1} v^T AA^T v$$

is linked with

$$w = \arg\min_{u^T\mathbf{1}=0} (v+u)^T AA^T(v+u) + v^T AA^T v \times u^T \begin{pmatrix} \frac{1}{v_1^2} & 0 & 0 \\ & ... & \\ 0 & 0 & \frac{1}{v_M^2} \end{pmatrix} u$$

But,

$$\nu_2 = \arg\min_{v\geq\mathbf{0}, v^T v\geq 1} v^T AA^T v$$

is linked with

$$w = \arg\min_{u^T v=0} (v+u)^T AA^T(v+u) + v^T AA^T v \times u^T \begin{pmatrix} \frac{1}{v_1^2} & 0 & 0 \\ & ... & \\ 0 & 0 & \frac{1}{v_M^2} \end{pmatrix} u$$

(then one could normalize by $v^T v$)
and

$$\nu_\infty = \arg\min_{v\geq\mathbf{1}} v^T AA^T v$$

is linked with

$$w = \arg\min_{u} (v+u)^T AA^T(v+u) + v^T AA^T v \times u^T \begin{pmatrix} \frac{1}{(v_1-1)^2} & 0 & 0 \\ & ... & \\ 0 & 0 & \frac{1}{(v_M-1)^2} \end{pmatrix} u$$

In three case, if $A_i A^T \nu < 0$ one could increase ν_i while reducing $\nu^T AA^T \nu$. ν_∞ seems more interesting as it seems linked to some vertex: ν_∞ seems to be the solution of $v_I = \mathbf{1}$, $A_J A^T v = \mathbf{0}$ (this seems to prove that ν is polynomial in the binary size of A - this is not that trivial because ν is **not** an eigen value). Also the ∞ version does not require any normalization when performing perceptron like update. So, considering ν_∞ instead ν_1 should be interesting, yet, it seems not in preliminary numerical experiments - but this may lead to better theoretical properties. Also, ν_2 seems more elegant but may require normalization i.e. square root - so it is not interesting for the purpose of making easier arbitrary large number optimization.

References

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

[2] George B et. al. Dantzig. The generalized simplex method for minimizing a linear form under linear inequality restraints. In *Pacific Journal of MathematicsAmerican Journal of Operations Research*, 1955.

[3] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[4] Leonid Khachiyan. A polynomial algorithm for linear programming. *Doklady Akademii Nauk SSSR*, 1979.

[5] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Siam, 1994.

[6] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.