Enhanced Basic Procedures for the Projection and Rescaling Algorithm

David Huckleberry Gutman

Abstract

Using an efficient algorithmic implementation of Caratheodory's theorem, we propose three enhanced versions of the Projection and Rescaling algorithm's basic procedures each of which improves upon the order of complexity of its analogue in [Mathematical Programming Series A, 166 (2017), pp. 87-111].

1 Introduction

Peña and Soheili [5] propose a two-step *projection and rescaling algorithm*, which extends an algorithm by Chubanov [3], to solve the conic feasibility problem

$$\text{Find } x \in L \cap \mathbb{R}_{++}^n \tag{1}$$

where L subspace of $\mathbb{R}^n$ [3, 5]. Assuming the projection matrix, P_L, for L is available we can rewrite (1) as

$$\text{Find } x \text{ such that } P_L x > 0. \tag{2}$$

The projection and rescaling algorithm consists of two subprocedures:

1. *Basic Procedure (Projection)*: This procedure uses P_L to find a point in $L \cap \mathbb{R}_+^n$ provided this cone contains a deeply interior point. This is implemented via one of three schemes based on the Von Neumann/Perceptron algorithm.

2. *Rescaling*: Using the final iterate from the basic procedure, this step rescales $L \cap \mathbb{R}_+^n$ such that its interior points - provided $L \cap \mathbb{R}_+^n \neq \emptyset$ - become "deeper" in the interior of $\mathbb{R}_+^n$.

In this paper, we propose enhancements of each of the Von Neumann/Perceptron basic procedures. Our enhancements iteratively apply a technique used to prove Caratheodory's theorem. These enhanced procedures improve the complexity of the basic procedure from $O(n^4m)$ to $O(n^2m^3)$ operations when L has dimension m: a significant improvement when $m << n$.

Fundamentally, the basic procedure of Peña and Soheili adapts the Von Neumann and Perceptron procedures to iteratively reduce $\|P_L x\|_2$ on the standard n dimensional

simplex $\Delta_{n-1} := \{x \in \mathbb{R}^n : \|x\|_1 = 1, x \geq 0\}$ until either $P_L x > 0$ or $\|P_L x\|_2 \leq \frac{1}{3\sqrt{n}}\|x\|_\infty$. Thus, the basic procedure intends to approximately solve the subproblem

$$\min_{x \in \Delta_{n-1}} \|P_L x\|_2^2. \tag{3}$$

The convergence proofs in [5] depends on the observation that $\|x\|_\infty \geq \frac{1}{n}$ for all $x \in \Delta_{n-1}$. As such, the reasoning in [5] yields a faster rate provided we restrict the iterates of the Von Neumann/Perceptron schemes to proper faces of Δ_{n-1}. If $Q \in \mathbb{R}^{n \times m}$ is an orthonormal basis for L then $P_L = QQ^T$ and we may rephrase (3) as

$$\min_{x \in \Delta_{n-1}} \|Q^T x\|_2^2 = \min_{z \in \operatorname{conv}(Q^T)} \|z\|_2^2 \tag{4}$$

where $\operatorname{conv}(Q^T)$ denotes the convex hull of Q^T's columns. By Caratheodory's theorem, any point in $\operatorname{conv}(Q^T)$ can be written as convex combination of at most $m+1$ columns of Q^T. Our proposed enhancements apply this observation to ensure that each of the iterates is a convex combinations of no more than $m + 1$ columns of Q^T.

Our enhanced basic procedures iteratively reduce the objective (4) using a Von Neumann/Perceptron scheme which applies a modified version of the Incremental Representation Reduction (IRR) procedure of [1] at each iteration. When provided a point $z \in \operatorname{conv}(Q^T)$, the IRR outputs a new affinely independent, convex representation of the point z provided it already contains a sufficiently large set of affinely independent vectors in its support. Whereas the IRR operates in $O(m^3)$ time, our version operates in $O(m^2)$ time by allowing for vectors in the representation of x that have zero support.

This paper is organized as follows. Section 2 describes the mIRR, and proves important properties of it including its $O(m^2)$ complexity. Section 3 describes the limited support Von Neumann and Perceptron algorithms. Section 4 elaborates possible extensions.

2 Modified Incremental Representation Reduction

The heart of our improved basic procedure is a modified version of the Incremental Representation Reduction Procedure of [1]. This subprocedure iteratively applies the main technique used in standard proofs of Caratheodory's theorem [4]. To simplify notation, given a matrix $A \in \mathbb{R}^{m \times n}$ we define $\tilde{A}$ as the augmented matrix $\begin{bmatrix} 1 \dots 1 \\ A \end{bmatrix}$. If $B = [B(1), ..., B(k)] \subseteq \{1, ..., n\}$ is an ordered set of indices and $x \in \mathbb{R}^n$ we let $A_B = [A_{B(1)} ... A_{B(k)}] \in \mathbb{R}^{m \times k}$ where $A_{B(i)}$ denotes the $B(i)$-th column of A and $x_B = (x_{B(i)} ... x_{B(k)}) \in \mathbb{R}^k$ and $x_{B(i)}$ denotes the $B(i)$-th entry of x. Given a full column rank matrix $M \in \mathbb{R}^{k \times l}$, we let $M^\dagger$ denote its unique pseudoinverse, $(M^T M)^{-1} M^T$. This notation strongly mimics the notation used in [2] to present the revised Simplex method. The resemblance is entirely intentional; our method strongly resembles the revised Simplex method.

Theorem 1. *Suppose $B \subseteq \{1, ..., n\}$ is an ordered set of indices such that A_B consists of affinely independent columns and $\tilde{A}_B^\dagger$ is known. If $z = Ax = A_B x_B + A_j x_j$ for some $x \in \Delta_{n-1}$ and $j \in \{1, ..., n\} \backslash B$ then we can find $x^+ \in \Delta_{n-1}$, an ordered set of indices $B^+ \subseteq B' := \begin{bmatrix} B & j \end{bmatrix}$, and $\tilde{A}_{B^+}^\dagger$ such that $z = Ax = A_{B^+} x_{B^+}^+$ and A_{B^+} consists of affinely independent columns in $O(m^2)$ operations.*

Proof. There are two cases we must tackle:

1. $\tilde{A}_j \neq \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$: A_j is affinely *independent* of the columns of A_B, i.e. the matrix $\begin{bmatrix} A_B & A_j \end{bmatrix}$ has affinely independent columns.

2. $\tilde{A}_j = \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$: A_j is affinely *dependent* on the columns of A_B, i.e. the matrix $\begin{bmatrix} A_B & A_j \end{bmatrix}$ has affinely independent columns.

Determining the equality of $\tilde{A}_j$ and $\tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$ requires vector-matrix multiplication, an $\mathcal{O}(m^2)$ operation.

Case 1 ($\tilde{A}_j \neq \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$): In this case, let $B^+ = B'$ and $x^+ = x$. We claim that $\tilde{A}_{B^+}^\dagger$ is given by the formula

$$\tilde{A}_{B^+}^\dagger = \begin{bmatrix} \tilde{A}_B^\dagger \\ 0 \end{bmatrix} - \begin{bmatrix} \tilde{A}_B^\dagger \tilde{A}_j \\ -1 \end{bmatrix} \frac{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger)}{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \tilde{A}_j}. \tag{5}$$

The quantity $\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \tilde{A}_j = \|\tilde{A}_j - \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j\|^2$ is non-zero by hypothesis and thus the expression on the right is well defined. It suffices to verify that right multiplication of the right hand side of (5) by $\tilde{A}_{B^+} = \begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix}$ yields the identity matrix. We compute

$$\begin{bmatrix} \tilde{A}_B^\dagger \\ 0 \end{bmatrix} \begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix} = \begin{bmatrix} \tilde{A}_B^\dagger \tilde{A}_B & \tilde{A}_B^\dagger \tilde{A}_j \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I_{k \times k} & \tilde{A}_B^\dagger \tilde{A}_j \\ 0 & 0 \end{bmatrix} \tag{6}$$

$$\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix} = \tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \tilde{A}_j \begin{bmatrix} 0 & 1 \end{bmatrix}. \tag{7}$$

Observe that the right hand side of (7) is non-zero since $\tilde{A}_j \neq \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$. Combining equations (6) and (7) yields

$$\left(\begin{bmatrix} \tilde{A}_B^\dagger \\ 0 \end{bmatrix} - \begin{bmatrix} \tilde{A}_B^\dagger \tilde{A}_j \\ -1 \end{bmatrix} \frac{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger)}{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \tilde{A}_j} \right) \begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix} = \begin{bmatrix} I_{k \times k} & \tilde{A}_B^\dagger \tilde{A}_j \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & \tilde{A}_B^\dagger \tilde{A}_j \\ 0 & -1 \end{bmatrix}$$

$$= I_{(k+1) \times (k+1)}$$

thus verifying our formula for $\tilde{A}_{B^+}^\dagger$. The formula (5) uses matrix addition and vector-matrix multiplication so it takes at most $\mathcal{O}(m^2)$ operations.

Case 2 ($\tilde{A}_j = \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$): Let $u = \begin{bmatrix} \tilde{A}_B^\dagger \tilde{A}_j & -1 \end{bmatrix}$, $\theta^* = \max_{i:u_i < 0} \frac{x_{B'(i)}}{u_i}$, $x_{B'}^+ = x_{B'} + \theta^* u$, and $x_{(B')^c}^+ = 0$. We must show that $x^+ \in \Delta_{n-1}$. By hypothesis, u is the solution to the system

$$\begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix} u = 0$$

3

because $\tilde{A}_j = \tilde{A}_B \tilde{A}_B^\dagger \tilde{A}_j$. Hence, $\sum_{i=1} u_i = 0$ which implies

$$\sum_{i=1}^{k+1} x_{B'(i)}^+ = \sum_{i=1}^{k+1} x_{B'(i)} = 1.$$

Moreover, the definition of θ^* ensures $x^+ \geq 0$ completing our proof that $x^+ \in \Delta_{n-1}$.

Next, we construct B^+ and $\tilde{A}_{B^+}^\dagger$. Let i^* the smallest index such that $\frac{x_{B'(i^*)}}{u_i} = \theta^*$ and $u_i < 0$. By construction, θ^* ensures $x_{B^+(i^*)}^+ = 0$. We now have two subcases: $B(i^*) = j$ and $B(i^*) \neq j$. In the first case, let $B^+ = B$ and $\tilde{A}_{B^+}^\dagger = \tilde{A}_B^\dagger$. By hypothesis, $A_{B^+} = A_B$ consists of affinely independent columns. In the second case, let $B^+(i) = B(i)$ for $i \neq i^*$ and $B^+(i^*) = j$. We must show that A_{B^+} consists of affinely independent columns. Assume for the sake of contradiction that it does not. Then there must exist some $w \in \mathbb{R}^{k+1}$ such that $w_{i^*} = 0$ and

$$0 = \tilde{A}_{B^+} w - \tilde{A}_j = \tilde{A}_B w - \tilde{A}_j$$

but this implies that

$$\tilde{A}_B \left(w - \tilde{A}_B^\dagger \tilde{A}_j \right) = \tilde{A}_j - \tilde{A}_j = 0.$$

By affine independence of the columns of A_B we determine that

$$w - \tilde{A}_B^\dagger \tilde{A}_j = 0 \Leftrightarrow w = \tilde{A}_B^\dagger \tilde{A}_j$$

so that the i^*-th entry of $\tilde{A}_B^\dagger \tilde{A}_j$, which is precisely u_{i^*}, is zero: a contradiction. Thus, the columns of A_{B^+} are affinely independent. Finally, we prove that it is possible to derive A_{B^+} in $\mathcal{O}(m^2)$ operations in this second case. Form the augmented matrix $\left[\begin{array}{c|c} \tilde{A}_B^\dagger & \tilde{A}_B^\dagger \tilde{A}_j \end{array} \right]$. Add to each row a multiple of the i^*-th row to make the last column equal to the unit vector e_{i^*}. The first $|B|$ columns of the resultant matrix are $\tilde{A}_{B^+}^\dagger$. This requires no more than $\mathcal{O}(m^2)$ operations since at most m row operations are required. $\qquad\square$

The proof of this theorem immediately yields our core algorithm.

Algorithm 1 Modified Incremental Representation Reduction Procedure (mIRR)

1: Input: An ordered set of indices $B = [B(1), ..., B(k)] \subseteq \{1, ..., n\}$ such that A_B is a matrix with affinely independent columns, $\tilde{A}_B^\dagger$, $j \notin B$, and $y = Ax = A_B x_B + A_j x_j$ for some $x \in \Delta_{n-1}$ with $x_{[B,j]^c} = 0$.

2: Compute $u' = \tilde{A}_B^\dagger \tilde{A}_j$ and $u = \begin{bmatrix} (u')^T & -1 \end{bmatrix}$. If $\tilde{A}_B u' \neq \tilde{A}_j$ then the columns of $\begin{bmatrix} \tilde{A}_B & \tilde{A}_j \end{bmatrix}$ are affinely independent. In this case, output $x^+ = x$, $B^+ = B \cup \{j\}$, and

$$\tilde{A}_{B^+}^\dagger = \begin{bmatrix} \tilde{A}_B^\dagger \\ 0 \end{bmatrix} - \begin{bmatrix} \tilde{A}_B^\dagger \tilde{A}_j \\ 1 \end{bmatrix} \frac{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger)}{\tilde{A}_j^T (I - \tilde{A}_B \tilde{A}_B^\dagger) \tilde{A}_j}.$$

to complete the procedure. Otherwise, proceed to the next step.

3: Let $\theta^* = \max_{i:u_i<0} \frac{x_i}{u_i}$, i^* be the smallest index that for which θ^* is achieved, and

$$x_{[B,j]}^+ = x_{[B,j]} + \theta^* u, \quad x_{[B,j]^c}^+ = 0.$$

If $i^* = j$ then output x^+, $B^+ = B$, and $\tilde{A}_{B^+} = \tilde{A}_B$ to complete the procedure. Otherwise, proceed to the next step.

4: Let $B^+ = [B(1), ..., B(i^* - 1), j, B(i^*), ..., B(k)]$. Form the $|B| \times (|B| + 1)$ matrix $\begin{bmatrix} \tilde{A}_B^\dagger & u' \end{bmatrix}$. Add to each row a multiple of the i^*-th row to make the last column equal to the unit vector e_{i^*}. The first $|B|$ columns of the resultant matrix are $\tilde{A}_{B^+}^\dagger$. Output B^+, x^+ and $\tilde{A}_{B^+}^\dagger$.

The following is an easy corollary of theorem 1.

Corollary 1. *The mIRR produces an affinely independent representation, $y = Ax = A_{B^+} x_{B^+}$ with $x \in \Delta_{n-1}$ and $x_{(B^+)^c} = 0$, of the input point $y = Ax$ and the pseudoinverse $\tilde{A}_{B^+}^\dagger$ in $\mathcal{O}(m^2)$ operations.*

3 Limited Support Basic Procedures

In this section we propose each of our modified basic procedures. Recall that we assume the availability of an orthonormal basis for L and that $Q \in \mathbb{R}^{n \times m}$ is the matrix whose columns are these basis vectors. The convergence results of [5] for the original basic procedures depend upon the maximum size of the support of the iterates. If it were possible to ensure that the iterates maintained affinely independent support then the support would have maximum size $m + 1$. This is the crux of our enhanced procedures and the mIRR enables us to do this. Our enhanced procedures start with a single column of the matrix Q^T. Then, until the stopping condition is reached, they take a Von-Neumann/Perceptron-like step - which may increase the size of the support by at most one - followed by an application of mIRR to ensure the support remains affinely independent. We will let $\{q_i\}_1^n$ denote the columns of Q^T and we use P in place of P_L.

The first two schemes, the Limited Support Von Neumann and Limited Support Perceptron, are subsumed in the following framework which we call the *Limited Support Scheme* (LSS). Each of these procedures is an enhancement of those found in [5] using mIRR.

Algorithm 2 Limited Support Scheme

$x_0 = e_1$, $z_0 = Q^T x_0 = q_1$, $B_0 = \{1\}$, $\tilde{Q}^\dagger_{B_0} = \frac{1}{\|\tilde{q}_1\|}\tilde{q}_1^T$, $t = 0$

while $Px_t > 0$ and $\|(Px_t)^+\| \geq \frac{1}{3\sqrt{n}}\|x_t\|_\infty$ **do**

    Let $j = \text{argmin}_{i \in [n]} \langle q_i, z_t \rangle$

    $x'_{t+1} = x_t + \theta_t(e_j - x_t)$

    $z_{t+1} = Q^T x'_{t+1}$

    if $j \notin B_t$ **then**

        $(x_{t+1}, B_{t+1}, \tilde{Q}^\dagger_{B_{t+1}}) = mIRR(x'_{t+1}, B_t, j, \tilde{Q}^\dagger_{B_t}, Q_j)$

    else

        $x_{t+1} = x'_{t+1}$, $B_{t+1} = B_t$, $\tilde{Q}^\dagger_{B_{t+1}} = \tilde{Q}^\dagger_{B_t}$

    end if

    $t = t + 1$

end while

If $\theta_t = \frac{1}{t+1}$ then the resulting procedure is referred to as the *Limited Support Perceptron Scheme* (LSP). If θ_t is determine by an exact line search then the resulting procedure is referred to as the *Limited Support Von Neumann Scheme* (LSVN).

Proposition 1. *The following hold for algorithms LSP and LSVN:*

1. *For all $t \geq 0$ such that LSP or LSVN have not halted, $\|z_t\|^2 \leq \frac{1}{t}$.*

2. *The stopping condition, $Px_t > 0$ or $\|(Px)^+\| \leq \frac{\|x_t\|_\infty}{3\sqrt{n}}$, occurs in at most $9(m+1)^2 n$ iterations.*

3. *LSP and LSVN require $O(m^4 n)$ arithmetic operations.*

Proof. Part 1 of our proposition is known from [5].

2. $|\{i \in \{1, ..., n\} : x_i > 0\}| \leq |B_t| \leq m + 1$ throughout the algorithm since the columns representing x_t are affinely independent. Thus, $\|x_t\|_\infty \geq \frac{1}{m+1}$ since $x_t \in \Delta_m$. This implies that $\frac{1}{3\sqrt{n}}\|x\|_\infty \geq \frac{1}{3(m+1)\sqrt{n}}$. As $\|(Px)^+\| \leq \|Px\|$, we conclude from 1 that the one of the two stopping conditions occurs by $t = 9(m+1)^2 n$.

3. By part 1, LSP and LSVN terminate in at most $t = 9(m+1)^2 n$ main iterations. Each of the operations besides mIRR requires at most nm operations while theorem 1 states mIRR requires $O(m^2)$ operations. Since $m^2 \leq nm$, we conclude each iteration has computational cost $O(nm)$. Thus, the number of required arithmetic operations for LSP and LSVN is $O(m^3 n^2)$.

$\square$

3.1 Limited Support Von Neumann with Away Steps Scheme

Here we propose a limited support variation of the Von Neumann with Away Steps scheme proposed in [5]. This procedure is essentially the same as above except that it allows for "away" directions.

Algorithm 3 Limited Support Von Neumann with Away Steps Scheme (LSVN)

$x_0 = e_1$, $z_0 = Q^T x_0 = q_1$, $B_0 = \{1\}$, $\tilde{Q}^{\dagger}_{B_0} = \frac{1}{\|\tilde{q}_1\|} \tilde{q}_1^T$, $t = 0$

while $Px_t > 0$ and $\|(Px_t)^+\| \geq \frac{1}{3\sqrt{n}} \|x_t\|_{\infty}$ **do**

   Let $j = \mathrm{argmin}_{i \in [n]} \langle q_i, z_t \rangle$, $k = \mathrm{argmax}_{i \in [n]} \langle q_i, z_t \rangle$

   if $\|z_t\|^2 - \langle q_j, z_t \rangle > \langle q_k, z_t \rangle - \|z_t\|^2$ **then**

      (Regular Step) $a := e_j - x_t$; $\theta_{max} = 1$

   else

      (Away Step) $a := x_t - e_k$; $\theta_{max} = \frac{(x_t)_j}{1 - (x_t)_j}$

   end if

   $\theta_t = \mathrm{argmin}_{\theta \in [0, \theta_{max}]} \|P(x_t + \theta a)\|^2 = \min\left\{\theta_{max}, -\frac{\langle x_t, Pa \rangle}{\|Pa\|^2}\right\}$

   $x'_{t+1} = x_t + \theta_t a$

   $z_{t+1} = Q^T x'_{t+1}$

   if $j \notin B_t$ and a regular step is taken **then**

      $(z_{t+1}, x_{t+1}, B_{t+1}, \tilde{Q}^{\dagger}_{B_{t+1}}) = mIRR(z_{t+1}, x'_{t+1}, B_t, j, \tilde{Q}^{\dagger}_{B_t}, Q_j)$

   else

      $x_{t+1} = x'_{t+1}$, $B_{t+1} = B_t$, $\tilde{Q}^{\dagger}_{B_{t+1}} = \tilde{Q}^{\dagger}_{B_t}$

   end if

   $t = t + 1$

end while

Proposition 2. *The following hold for algorithm LSVNA:*

1. *For all $t \geq 0$ such that LSVNA has not halted, $\|z_t\|^2 \leq \frac{1}{t}$.*

2. *The stopping condition, $Px_t > 0$ or $\|(Px)^+\| \leq \frac{\|x_t\|_{\infty}}{3\sqrt{n}}$, occurs in at most $9(m + 1)^2 n$ iterations.*

3. *LSVNA requires $O(m^4 n)$ arithmetic operations.*

Proof. The proof of the first part of the proposition is known from [5]. The remaining parts follows from similar reasoning to that in the proof of proposition 1. $\square$

4 Extensions

The original paper [5] extends the problem (1) to the semidefinite cone and the more general symmetric cone case. It is maybe possible to extend our enhanced algorithms to these settings. The primary obstacle will be defining the systems of equations necessary to mIRR.

References

[1] Amir Beck and Shimrit Shtern. Linearly convergent away-step conditional gradient for non-strongly convex functions. *Mathematical Programming*, 164(1):1–27, 2017.

[2] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

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

[4] J. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of convex analysis*. Springer Science & Business Media, 2012.

[5] Javier Peña and Negar Soheili. Solving conic systems via projection and rescaling. *Mathematical Programming*, 166(1):87–111, 2017.