A simple and numerical stable algorithm for solving the cone projection problem based on a Gram-Schmidt process

Demetris T. Christopoulos[1]

[1]*Department of Economics, National and Kapodistrian University of Athens**

(Dated: July 12, 2012)

Abstract

We are presenting a simple and numerical stable algorithm for the solution of the cone projection problem which is suitable for relative small data sets and for simulation purposes needed for convexity tests. Not even one pseudo-inverse matrix is computed because of a proper Gram-Schmidt orthonormalization process that is used.

Keywords: cone projection algorithm, convexity constraints, convexity test simulations, Gram-Schmidt.

THE CONE PROJECTION PROBLEM

We have the data set $(x_i, \phi_i), i = 1, 2, \ldots, n$ which has emerged from a convex function f at least $C^{(2)}[x_1, x_n]$ by the process:

$$\phi_i = f(x_i) + \epsilon_i,\ \epsilon \sim iid(0, \sigma^2\, I_n) \tag{1}$$

We want to find the vector y that has the smallest euclidean distance from ϕ subject to the requirement of convexity $A\, y \geq 0$, thus we have to solve the next primal optimization problem:

$$\begin{gathered} \min \left\{ \sum_{i=1}^{n} \left(y_i - \phi_i\right)^2 = (y-\phi)^T\, (y-\phi) \right\} \\ \text{subject to: } (-A\, y) \leq 0 \end{gathered} \tag{2}$$

There are two equivalent versions for the matrix A of the convexity inequalities constraints. The first one is is to observe that we have strict inequalities:

$$x_1 < x_2 < \cdots < x_n$$

so starting from the definition of convexity we proceed to the inequalities:

$$\frac{y_{i+2} - y_{i+1}}{x_{i+2} - x_{i+1}} \geq \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

$$\left(y_{i+2} - y_{i+1}\right)\left(x_{i+1} - x_i\right) \geq \left(y_{i+1} - y_i\right)\left(x_{i+2} - x_{i+1}\right)$$

$$\left(x_{i+2} - x_{i+1}\right) y_i + \left(x_i - x_{i+2}\right) y_{i+1} + \left(x_{i+1} - x_i\right) y_{i+2} \geq 0 \tag{3}$$

By constructing now all the above inequalities for $i = 1, 2, \ldots, n-2$ we have formulated the matrix $A^{(i)}$.

$$A^{(i)} = \begin{pmatrix} x_3 - x_2 & x_1 - x_3 & x_2 - x_1 & 0 & \cdots & 0 \\ 0 & x_4 - x_3 & x_2 - x_4 & x_3 - x_2 & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & x_n - x_{n-1} & x_{n-2} - x_n & x_{n-1} - x_{n-2} \end{pmatrix} \tag{4}$$

The second way is obtained if we have equal spaced x_i-data. Then it is easy to eliminate the same positive quantity $\Delta x = x_{j+1} - x_j$ from all inequalities:

$$\left(x_{i+2} - x_{i+1}\right) y_i + \left(x_i - x_{i+2}\right) y_{i+1} + \left(x_{i+1} - x_i\right) y_{i+2} \geq 0$$

$$\left(\Delta x\right) y_i - \left(2\Delta x\right) y_{i+1} + \left(\Delta x\right) y_{i+2} \geq 0$$

$$y_i - 2y_{i+1} + y_{i+2} \geq 0 \tag{5}$$

again with $i = 1, 2, \ldots, n-2$ and create the matrix $A^{(ii)}$.

$$A^{(ii)} = \begin{pmatrix} 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & \cdots & 1 & -2 & 1 \end{pmatrix} \tag{6}$$

Lemma .1. *The polar component ρ^* of the vector decomposition $\phi = y^* + \rho^*$ with y^* the solution of problem*

2 is a linear combination of the negative rows of matrix A, while the coefficients are either zero (if the corresponding constraint is not binding - inactive) or positive (if the relevant constraint is binding - active).

Proof. The Lagrangian function of the problem and the first order condition for y can be written as:

$$L\left(y,\lambda\right)=\left(y-\phi\right)^{T}\ \left(y-\phi\right)+\lambda^{T}\ \left(-A\,y\right) \tag{7}$$

$$\begin{gathered}\frac{\partial L(y,\lambda)}{\partial y}=2\ \left(y-\phi\right)+\left(-A^{T}\right)\ \lambda=0\\ \text{or}\\ 2\ \left(y-\phi\right)+\left(-A^{T}\right)\ \lambda=0\\ \text{or}\\ y=\phi+\tfrac{1}{2}\,A^{T}\,\lambda\end{gathered} \tag{8}$$

So, for the optimal solution $\{y^{*},\lambda^{*}\}$ we have that it also holds:

$$\begin{gathered}y^{*}=\phi+\tfrac{1}{2}\,A^{T}\,\lambda^{*}\\ \text{or}\\ \phi-y^{*}=-\tfrac{1}{2}\,A^{T}\,\lambda^{*}\\ \text{or}\\ \rho^{*}=\left(-A^{T}\right)\ \tfrac{\lambda^{*}}{2}\\ \text{or}\\ \rho^{*}=\left(-A^{T}\right)\ \hat{\lambda}^{*}\end{gathered} \tag{9}$$

Thus the representation of the polar component of the data vector, following the definitions of [1] and [3] in the basis of the negative rows of A is half the Lagrange coefficient vector of the optimization problem 2. The coefficients are zero or positive if the corresponding constraint is inactive or active respectively, due to Karush Kuhn Tucker complementarity slackness conditions. □

A NUMERICAL STABLE GEOMETRIC ALGORITHM FOR CONE PROJECTION

An illustrative example

Example .1. *Let' s start with a common convex function:*

$$f(x)=x^{2}\,,x\in[0,1] \tag{10}$$

Let the vectors $x=(0,\frac{1}{2},1,\frac{3}{2},2)$ *and* $\phi=(0,\frac{1}{2},\frac{5}{2},\frac{15}{4},4)$ *as presented at Figure 1 where we have drawn also the chord connecting* (x_1,ϕ_1) *and* (x_5,ϕ_5)*. If our data was convex then all* (x_i,ϕ_i) *should lie above the chord, so clearly we have not convexity here.*

The demand for convexity takes the form of the next inequality constraints, using matrix $A^{(ii)}$ *because of the*

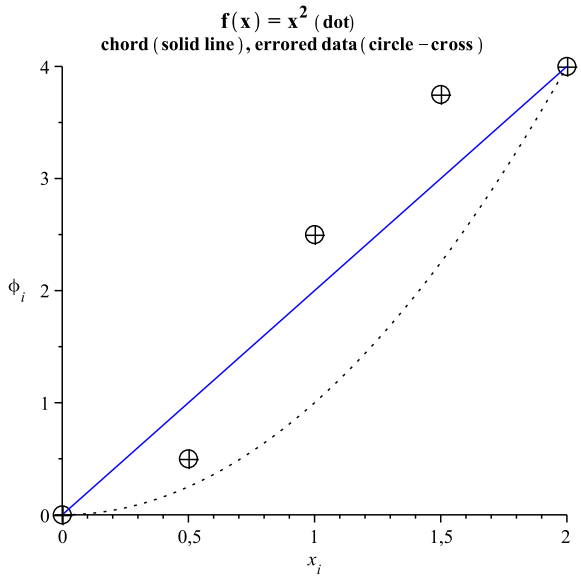

FIG. 1: The statement of the convex projection problem for $n=5$

equal spaced x_i*:*

$$\begin{gathered}A\,y\geq 0\\ A=\begin{pmatrix}1&-2&1&0&0\\0&1&-2&1&0\\0&0&1&-2&1\end{pmatrix}\end{gathered} \tag{11}$$

We define the matrix:

$$R=-A=\begin{pmatrix}-1&2&-1&0&0\\0&-1&2&-1&0\\0&0&-1&2&-1\end{pmatrix}$$

We pre-multiply vector ϕ *by R:*

$$R\,\phi=\begin{pmatrix}-\frac{3}{2}\\ \frac{3}{4}\\ 1\end{pmatrix}$$

If all components of the result were negative, then for the matrix A it should hold $A\,\phi\geq 0$*, so our data should be convex. So, if we seek for the greatest deviation from convexity then it is natural to pick the component that is the greatest positive. This is compatible with (i) deviation from convexity and with (ii) Lemma .1.*
Here we observe the greatest entrance to be the 3rd one, so we pick up the 3rd row of R as the best chance to obtain a component of the polar vector and proceed by

taking the orthogonal projection of ϕ onto that row:

$$R_1 = r_3^T = \begin{pmatrix} 0 \\ 0 \\ -1 \\ 2 \\ -1 \end{pmatrix}$$

$$\mu_1 = \left(\frac{\langle \phi, r_3 \rangle}{\langle r_3, r_3 \rangle}\right) = \left(\frac{1}{6}\right) = \left(0.166666\ldots\right) = \left(0.1\overline{6}\right)$$

$$\rho_1 = R_1\,\mu_1 = \begin{pmatrix} 0 \\ 0 \\ -\frac{1}{6} \\ \frac{1}{3} \\ -\frac{1}{6} \end{pmatrix}$$

$$y_1 = \phi - \rho_1 = \begin{pmatrix} 0 \\ \frac{1}{2} \\ \frac{8}{3} \\ \frac{41}{12} \\ \frac{25}{6} \end{pmatrix}$$

$$R_1\,y_1 = \begin{pmatrix} -\frac{5}{3} \\ \frac{17}{12} \\ 0 \end{pmatrix}$$

Now the greatest entry is the 2nd one, so we pick up the 2nd row of R and continue by taking the matrix with the 2nd and 3rd rows of R. It is important to notice that we are always sorting our indices in ascending order.

$$R_2 = \begin{pmatrix} r_2^T & r_3^T \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -1 & 0 \\ 2 & -1 \\ -1 & 2 \\ 0 & -1 \end{pmatrix}$$

Now there exist two ways for projecting our data ϕ on the two columns of R_2:

1. *The traditional way, i.e. the OLS estimator, which involves the pseudo inverse matrix and implies many numerical instabilities*
2. *The new proposed way of taking the projection on the orthonormal base produced from them via the Gram-Schmidt procedure.*

We choose 2^{nd} way and first construct for R_2 with Gram-Schmidt the matrix with orthonormal columns:

$$V = \begin{pmatrix} 0 & 0 \\ -\frac{\sqrt{6}}{6} & -\frac{\sqrt{30}}{15} \\ \frac{\sqrt{6}}{3} & \frac{\sqrt{30}}{30} \\ -\frac{\sqrt{6}}{6} & \frac{2\sqrt{30}}{15} \\ 0 & -\frac{\sqrt{30}}{10} \end{pmatrix}$$

Then we just take the projections of ϕ on the two columns of V:

$$\mu_2 = V^T\,\phi = \begin{pmatrix} \frac{\sqrt{6}}{8} \\ \frac{3\sqrt{30}}{20} \end{pmatrix} = \begin{pmatrix} 0.3061862179 \\ 0.8215838362 \end{pmatrix}$$

The reader who is familiar with the active set methodology has to notice that our vector is not identical anymore to the λ vector of that method, because of the orthonormalization process. This is the cost for the numerical stabilization of our algorithm.We continue executing our algorithm:

$$\rho_2 = V\,\mu_2 = \begin{pmatrix} 0 \\ -\frac{17}{40} \\ \frac{2}{5} \\ \frac{19}{40} \\ -\frac{9}{20} \end{pmatrix}$$

$$y_2 = \phi - \rho_2 = \begin{pmatrix} \frac{37}{40} \\ \frac{21}{10} \\ \frac{131}{40} \\ \frac{89}{20} \end{pmatrix}$$

$$R_2\,y_2 = \begin{pmatrix} -\frac{1}{4} \\ 0 \\ 0 \end{pmatrix}$$

We observe that there exist no polar edge vector to be inserted in our algorithm, so we exit with the solutions:

$$\rho^* = \rho_2 = \begin{pmatrix} 0 \\ -\frac{17}{40} \\ \frac{2}{5} \\ \frac{19}{40} \\ -\frac{9}{20} \end{pmatrix}$$

$$y^* = y_2^* = \begin{pmatrix} \frac{37}{40} \\ \frac{21}{10} \\ \frac{131}{40} \\ \frac{89}{20} \end{pmatrix}$$

We check again our solution:

$$A\,y^* = \begin{pmatrix} \frac{1}{4} \\ 0 \\ 0 \end{pmatrix}$$

$$\langle y^*, \rho^* \rangle = 0$$

It is expected to find that we will have two distinct lines for our cone projection plot and the second line has to be the OLS line for the set $\{(x_i, \phi_i), i = 2, 3, 4, 5\}$, because only then we have a sequence of vanishing constraints. This fact is easily observed at Figure 2.

The algorithm

By increasing the dimension of our problem until a rather big n we continue to apply the same actions, i.e. we have established an *algorithm* for cone projection.

We start by testing if our data is convex, so there is no need for cone projection at all. If it is not convex, then we multiply it by the R matrix and seek for the maximum component and for its position. That direction is more

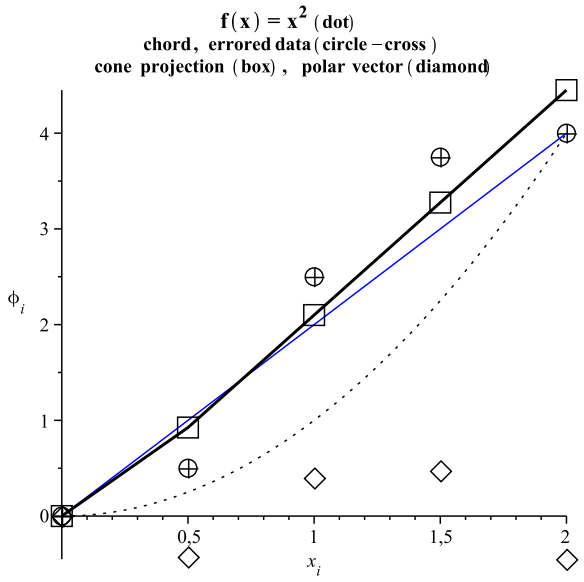

FIG. 2: The geometry of convex projection problem for $n = 5$ in xy-plot

probable to be an edge of the polar cone, so we find the projection of our data onto the i^{th} row of R matrix. Now we have found the first approximations of the vectors ρ and $y = \phi - \rho$. We multiply again this y with R ($b = R\,y$) and seek again for the maximum component and for its position. The new direction forms a set together with the previous one and we always sort the indices. The sorted indices construct the X matrix by taking the corresponding rows of R matrix as the columns of X. Then we apply the Gram-Schmidt orthonormalization procedure on the columns of X and construct the matrix V. Now our μ vector can be calculated and then we find the next ρ and y approximation. We continue our algorithm until we reach at least one of the next three termination criteria:

1. The algorithm is terminated if some b vector is 'practically' zero
2. The algorithm is terminated if there is no improvement in the value of b
3. The algorithm is terminated if next index i of $R-$row has already been chosen.

Finally we exit from the algorithm with the set of indices J, where we have that the convexity constraints are satisfied as equalities (the active set indices, but without calculating the corresponding Lagrange coefficients), the polar component ρ^* and the cone projection component y^* of our initial data ϕ. *We do not compute even one time any kind of pseudo-inverse matrix, which is the fundamental tool of every regression technique.* This is due to the use of Gram-Schmidt orthonormalization process in order to do our orthogonal projections. This makes the algorithm *numerical stable* for using it *for simulation purposes*: we can establish the cone projection solution for every random set of vectors. This cannot be done with the traditional OLS solution, because of the existence of almost singular matrices for floating point arithmetic computations. The pseudo-code of the Algorithm is presented below.

A Gram-Schmidt polar basis Cone Projection Algorithm

Find $y^* = \underbrace{argmin}_{y} \|y - \phi\|_2$

subject to $A\,y \geq 0$

INITIALIZE

$\{\epsilon_1, \epsilon_2, J = \{\}, R = -A, b = R\,\phi, b_{old} = b + \theta, \theta > 0\}$

- **IF** $\{b \geq 0\}$ **THEN** $\{\rho = 0, y = \phi\}$ **BREAK**
- **ELSE**

FIRST PROJECTION

$$\left\{ s = \underbrace{max}_{j=1,\ldots,n-2} \; b_j \;\; i = \underbrace{arg}_{j=1,\ldots,n-2} \; (b_j = s) \;\; J = sort\,(J \cup \{i\}) \right\}$$

$$\rho = project_{r_i}\,\phi$$

$$y = \phi - \rho, b = R\,y$$

NEXT PROJECTIONS

$$\left\{ s = \underbrace{max}_{j=1,\ldots,n-2} \; b_j \;\; i = \underbrace{arg}_{j=1,\ldots,n-2} \; (b_j = s) \;\; J = sort\,(J \cup \{i\}) \right\}$$

IF $\{s \leq \epsilon_1\}$ **THEN BREAK**

DO WHILE $\|b - b_{old}\|_1 \geq \epsilon_2$

$$\left\{ X = \left(r_{i_1}^T \ldots r_{i_k}^T\right),\; i_j \in J \right\}$$

$$\left\{ V = GramSchmidt(X) \right\}$$

$$\rho = project_V\,\phi$$

$$y = \phi - \rho, b = R\,y$$

$$\left\{ s = \underbrace{max}_{j=1,\ldots,n-2} \; b_j \;\; i = \underbrace{arg}_{j=1,\ldots,n-2} \; (b_j = s) \;\; J = sort\,(J \cup \{i\}) \right\}$$

- **IF** $\{s \geq \epsilon_1\}$ **THEN** $\{J = sort\,(J \cup \{i\})\}$
- **IF** $\{i \in J\}$ **THEN BREAK**
- **ELSE BREAK**

END DO

CHECK SOLUTION $|\langle y, \rho \rangle| \leq \epsilon_1$

RETURN $\{J, \rho, y\}$

We have developed our algorithm in four proper languages:

1. First in Maple symbolic algebra system, where with just one page code we are able to execute our algorithm in absolute accuracy by using rational num-

bers as input data.

2. Second in R suite, where we have floating point arithmetic, but it is an 'alter ego' for the statistician community.

3. Third in Matlab/Octave, for those who are familiar with the benefits of them.

4. Fourth in FORTRAN, one of the fastest ways to execute any numerical algorithm.

CONCLUSION

The presented algorithm for solving the cone projection problem is quite simple because:

- We don' t take care about the sign of Lagrange multipliers since we don' t compute them
- We just include one component of the polar basis every time

The algorithm is numerical stable for every kind of initial random vector ϕ because all projections are done via a Gram-Schmidt procedure and not with the common OLS pseudo-inverse matrix, which is very often close to singular for floating point arithmetic computations.
The algorithm is useful for convexity tests where we need to compute the weights of the weighted χ^2 or Beta distribution that emerges for the corresponding statistical test, see for example [2].

* Demetris T. Christopoulos, dchristop@econ.uoa.gr

[1] Rockafellar R. T. : *Convex analysis*, Princeton Landmarks in Mathematics, Princeton University Press, Princeton, NJ, (1997), ISBN 0-691-01586-4, reprint of the 1970 original, Princeton Paperbacks.
[2] Meyer M.C. : A test for linear versus convex regression function using shape-restricted regression, *Biometrika*, **90** (2003).
[3] Meyer M.C. : Consistency and power in tests with shape-restricted alternatives, *Journal of Statistical Planning and Inference*, **136** (2006).