Research project on random projections in LP

Reda Lahlou - Leder Yhivert Aguirre Ramirez
X2019
INF580 : Advanced Mathematical Programming

Advisor: Leo Liberti

Ecole Polytechnique
March 2022

Abstract

Random projections are random linear maps, sampled from appropriate distributions, namely, here, the gaussian distribution, that approximately preserve some geometrical invariants so that the approximation improves as the dimension of the space grows. According to the Johnson-Lindenstrauss lemma, there are random matrices with surprisingly few rows that approximately preserve pairwise Euclidean distances among a set of points.
Here, we're interested in the retrieval of the solution of a Linear Program (LP), through solving its projected dual. We then compare the solutions obtained from solving the LP, its projection, and then its projected dual.

Contents

1 Introduction

Here, as in [1], we start with a LP in standard form :

$$\mathcal{P} \equiv min\{c^T x | Ax = b \wedge x \geq 0 \wedge x \in X\} \tag{1}$$

(where $X = \mathbb{R}^m$) and obtain a randomly projected formulation under the RP $T \sim \mathcal{N}^{k\times n}(0, \frac{1}{\sqrt{k}})$ with the form :

$$T\mathcal{P} \equiv min\{c^T x | TAx = Tb \wedge x \geq 0 \wedge x \in X\} \tag{2}$$

Let $v(P)$ and $v(TP)$ be their respective optimal objective functions. If we make reasonable assumptions (that $feas(P)$ is bounded and that all optima of P satisfy $\Sigma_j x_j \leq \theta$ for some given $\theta > 0$ by the optimality projection theorem, for a given $\gamma > 0$,

$$v(P) - \gamma \leq v(TP) \leq v(P) \tag{3}$$

holds with arbitrarily high probability (the detailed proof can be found in [1]). However, the main issue is trying to retrieve a solution to P from TP. Indeed, while $Ax = b$ implies $TAx = Tb$, the reverse is not true according to the following :

Theorem : For $x \geq 0$ such that $TAx = Tb$, $Ax = b$ with probability zero. (4)

Hence, we can't get a solution for the original problem using the projected one.

Our idea (or rather, Pr Liberti's) is to solve P's projected dual. P's dual being $D \equiv max\{yb | yA \leq c\}$, a projected dual on $y \in \mathbb{R}^m$ can be derived :

$$TD \equiv max\{(yT^T)Tb | (yT^T)TA \leq c\} \tag{5}$$

Replacing $u = yT^T \in \mathbb{R}^k$, we get :

$$TD \equiv max\{u\bar{b} | u\bar{A} \leq c\} \tag{6}$$

where $\bar{b} = Tb$, $\bar{A} = TA$.
In theory, according to [2] :

- If the original dual is feasible, the projected dual is too

- We derive from the Johnson-Lindenstauss lemma approximation guarantees on the projected dual objective function
- If $\bar{u} \in \arg\text{opt}(\text{TD})$, let $\tilde{y} = \bar{u}T$

We then are able to solve TD, and then algorithmically find a candidate solution x' of P from $\tilde{y}$.

2 Methodology

We use a relaxation of the first-order Karush-Kuhn-Tucker (KKT) conditions to derive a candidate $\tilde{x}$ from $\tilde{y}$ for the problem P.[1] From now on, we will denote by TDP the problem of finding $\tilde{x}$ from $\tilde{x}$. Then, we test the performance of this solutions on random feasible instances. In particular, after solving P,TP and TDP, we calculate the feasibility error and the objective value with respect to P. It is important to note that we will generate multiple instances and projections for each configuration of the form (n, m, k) but the results will contain the average over the instances and the projections. In this way, the analysis must be carry out in a group perspective.

The model used to solve TDP using the relaxed KKT conditions is described below:

set	meaning
$I = \{0, \ldots, n-1\}$	constraints in P
$J = \{0, \ldots, m-1\}$	variables in P

parameter	meaning
$c\{J\}$	cost vector in P
$A\{I, J\}$	matrix of constraints in P
$b\{I\}$	constraints RHS in P
$\tilde{y}\{I\}$	projected solution of TD

[1] We do not apply the KKT conditions directly because, in general, $\tilde{y}$ will not be an optimal solution of D.

variable	meaning
$\tilde{x}\{J\}$	estimation of x
$s_p^+\{J\}$	positive deviation from primal KKT
$s_p^-\{J\}$	negative deviation from primal KKT
$s_d^+\{I\}$	positive deviation from dual KKT
$s_d^-\{I\}$	negative deviation from dual KKT

$$\begin{aligned}
\underset{s_p^+, s_p^-, s_d^+, s_d^-}{\text{minimize}} \quad & \sum_{j \in J} (s_p^+)_j + (s_p^-)_j + \sum_{i \in I} (s_d^+)_i + (s_p^-)_i \\
\text{subject to} \quad & \left(\sum_{i \in I} \tilde{y}_i A_{i,j} - c_j \right) \tilde{x}_j + (s_p^+)_j - (s_p^-)_j = 0, \; j \in J \\
& \tilde{y}_i \left(b_i - \sum_{j \in J} A_{i,j} \tilde{x}_j - \right) + (s_d^+)_i - (s_d^-)_i = 0, \; i \in I
\end{aligned}$$

3 Numerical results

For each $m \in \{300, 600, 900, 1500\}$ we consider $n = \left\lfloor \alpha m + \frac{1}{2} \right\rfloor$ where $\alpha \in \{0.9, 0.8, 0.7\}$. After that, for each n we take $k = \left\lfloor \beta n + \frac{1}{2} \right\rfloor$ where $\beta \in \{\frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}\}$. Then, for each tuple (n, m, k) we generate 5 instances and 5 projections per instance.

More specifically, the probability distribution assumed for each parameter is presented: $A \sim Uniform(-100, 100)$,$c \sim Uniform(0, 100)$ and, in order to get feasible instances we randomly generate $x' \sim Uniform(0, 100)$ and take $b = Ax$. For the random projections we have $T \sim Normal(0, \frac{1}{k})$.

All this part was made in a .run file written in AMPL. To solve this instances we create a .mod and a .run file with the implementation of the problems P,TP,TD and TDP, which is the name given to the model shown in the preceding section. Since we are dealing with instances of linear problems, we used the CPLEX solver. The output of the this phase is a file called *results* which contains columns:

variable	meaning
rows	n
columns	m
rows_proj	k
inst	instance index
P_status	CPLEX solve_result_num for P
TP_status	CPLEX solve_result_num for TP
TD_status	CPLEX solve_result_num for TD
TDP_status	CPLEX solve_result_num for TDP
P_feas_error	feasible error with respect to P[2]
TP_feas_error	$\|Ax-b\|_2^2$, x solution of TP
TD_feas_error	$\|Ax-b\|_2^2$, x solution of TPD
P_feas_error	$\|Ax-b\|_2^2$, x solution of TP
obj_val_P	value of P at x, x solution of P
obj_val_TP	value of TP at x, x solution of TP
obj_val_TDP	value of TDP at x, x solution of TDP

This data is then processed in a .ipynb file to get the figures below. The files mentioned in this section can be consulted on the repository [INF580-Random-Projections.

[2] This was calculated for validation purposes.

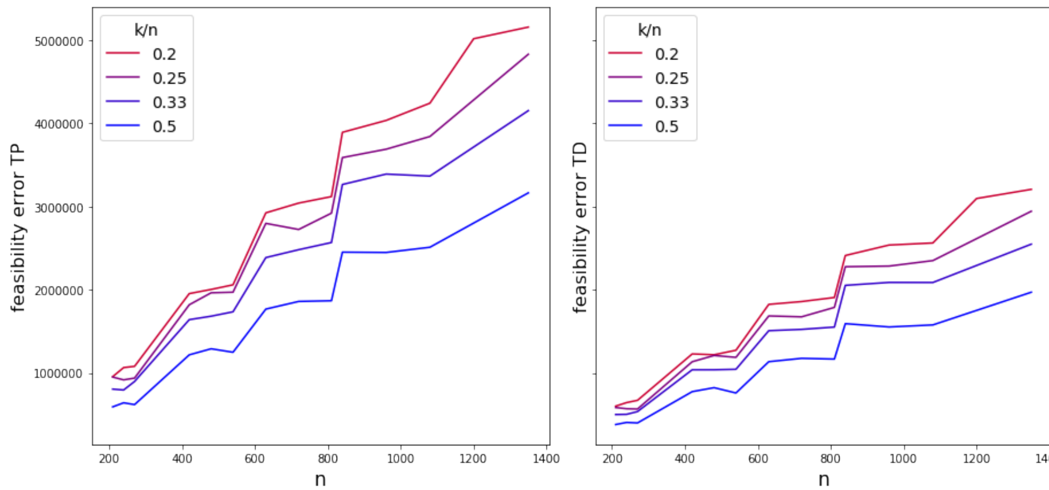

Figure 1: Average feasibility error of: (a) the solution of the primal-projected problem (TP) and the solution from the dual-projected problem (TD).

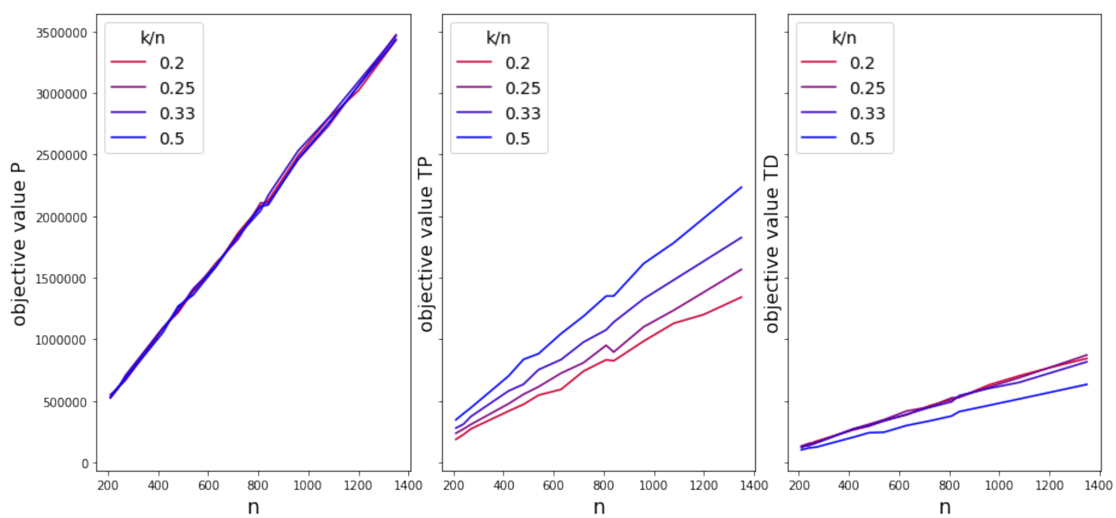

Figure 2: Average objective value of: (a) the solution of the primal-projected problem (TP) and the solution from the dual-projected problem (TD).

4 Conclusion

We notice that while the two feasibility error curves have similar profiles, the error for the dual problem seems to be significantly lower than for the primal problem. That certainly stems from the fact that dual functions provide lower bounds to the primal minimization formulations.

Furthermore, one must also note that Fig. 2 is actually the objective function value average taken over different projections and different instances. This is evidenced by the fact that the objective functions aren't straight lines. Unfortunately, this comes from a conception issue in our code : we generated instances for each m, n, and k instead of just for each m and n, and had to take the average instead of generating a lot of figures containing little information when taken separately.

Finally, we notice that while indeed, the average objective function value of TP gets closer to that of the original problem P when k/n increases, that is not the case for the dual problem TD. Which is why this study should be continued with higher-dimensional instances, to confirm this effect, to begin with, before we can start conjecturing on what might cause it.

References

1 Leo Liberti, Advanced Mathematical Programming, Lecture Notes.

2 Claudia D'Ambrosio, Leo Liberti, Pierre-Louis Poirion, Ky Vu, Random projections for quadratic programs, Mathematical Programming B, 183:619-647, 2020.