https://doi.org/10.2352/ISSN.2470-1173.2021.10.IPAS-226

Projection methods for finding intersection of two convex sets and their use in signal processing problems

Zuzana Bílková, Michal Šorel; The Czech Academy of Sciences, Institute of Information Theory and Automation, Prague, Czech Republic; Charles University in Prague, Prague, Czech republic

Abstract

Finding a point in the intersection of two closed convex sets is a common problem in image processing and other areas. Projections onto convex sets (POCS) is a standard algorithm for finding such a point. Dykstra's projection algorithm is a well known alternative that finds the point in the intersection closest to a given point. Yet another lesser known alternative is the alternating direction method of multipliers (ADMM) that can be used for both purposes. In this paper we discuss the differences in the convergence of these algorithms in image processing problems. The ADMM applied to finding an arbitrary point in the intersection is much faster than POCS and any algorithm for finding the nearest point in the intersection.

Introduction

A common problem in diverse areas of mathematics, physics and computer sciences is to find a point in the intersection of two closed convex sets. The problem appears in two variants, the *feasibility problem* with the aim to find an arbitrary point in the intersection of two closed convex sets; and the *best approximation problem* of finding the projection of a given point to the intersection, i.e. finding the point in the intersection closest to the given point.

These problems are important on their own but we often find them as subproblems in constrained optimization, in particular, in *projection methods* (interpretable as operator splitting methods) that combine smooth optimization with projections on a set of constraints [12]. The best known projection method is the projected gradient method. Its insufficient speed of convergence was improved in algorithms such as FISTA [5], the augmented Lagrangian method [16] and its extensions [13]. There are many applications in various fields, for example mechanics [18], traffic theory [17], game theory [8] and others. Projection methods have become a standard tool also in signal and image processing after their success in total variation denoising [10], deblurring, MRI reconstruction [1], and compressive sensing in general [2]. They are often applied to smooth problems constrained by sets with closed-form or otherwise fast projection such as halfspaces [4], spheres (unit norm sets) and many others [7]. These methods gained popularity expecially for the simplicity of implementation compared to for example interior-point methods and good convergence.

If the constraint is given as an intersection of two sets, there is usually no closed-form formula for the projection. There are two classical projection-based algorithms solving this particular problem. For the feasibility problem, we can use the alternating projection method, also known as the projections onto convex sets (POCS). It has been rediscovered numerous times [3]. The best approximation problem is often solved by Dykstra's projection algorithm [15].

As an alternative, the general projection methods such as ADMM can be used to compute both the feasibility and best approximation problems. Even though the number of projections can be high, they may be useful when we have an efficient method for computing the projections [9]. Rate of convergence is discussed in [14]. One particular advantage of the ADMM algorithm [16] is that it allows for partial updates, which means that we can run only several iterations in the subproblem and still have provable and in practice fast convergence. A similar algorithm would result from [13].

The main purpose of our paper is to compare efficiency of the POCS and Dykstra's algorithms with the algorithms based on ADMM. We also illustrate the fact that although the feasibility problem can be in theory solved using the best approximation problem algorithms, this is usually not a good idea because the latter are in practice much slower.

Alternating projection method

The simplest projection-based method for the feasibility problem (existence of intersection) is the alternating projection method. Let us denote the sets as $A \in \mathbb{R}^n$ and $B \in \mathbb{R}^n$, where $n \in \mathbb{N}$, and projections as P_A and P_B. The algorithm starts with an arbitrary value $y_0 \in X = \mathbb{R}^n$ and alternately projects on A and B :

$$x_{k+1} = P_A(y_k), \tag{1}$$

$$y_{k+1} = P_B(x_{k+1}), \tag{2}$$

where $k \in \mathbb{N}$. This generates sequences $(x_k)_{k\geq 1} \in A$ and $(y_k)_{k\in\mathbb{N}} \in B$. If the intersection of A and B is nonempty, the sequences $(x_k)_{k\geq 1}$ and $(y_k)_{k\in\mathbb{N}}$ both converge to a point in the intersection [11]. The method is illustrated in Figure 1.

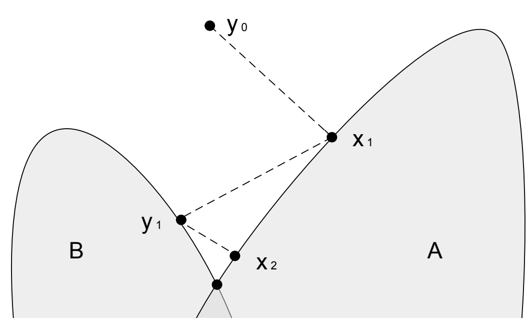

Figure 1. *Illustration of the alternating projection method.*

Dykstra's projection algorithm

Dykstra's projection algorithm [15] is a classical projection-based algorithm solving efficiently the best approximation problem. Roughly speaking, it iterates by projections in a clever way such that the resulting point is a projection of the starting point on the intersection, i.e. the result is the nearest point to the starting point in the intersection. Unlike the alternating projection method, there are intermediate steps. Complete discussion about this type of methods can be found in [14], convergence is proved in [12].

Dykstra's algorithm generates sequences $(x_k)_{k\geq 1}$, $(y_k)_{k\in\mathbb{N}}$, $(p_k)_{k\in\mathbb{N}}$ and $(q_k)_{k\in\mathbb{N}}$ as follows: Set $y_0 \in X$, $p_0 := 0$, $q_0 := 0$, and for every $k \in \mathbb{N}$ compute

$$
\begin{aligned}
x_{k+1} &= P_A(y_k + p_k), &(3)\\
p_{k+1} &= p_k + y_k - x_{k+1}, &(4)\\
y_{k+1} &= P_B(x_{k+1} + q_k), &(5)\\
q_{k+1} &= q_k + x_{k+1} - y_{k+1}. &(6)
\end{aligned}
$$

The sequence $(x_k)_{k\geq 1}$ converges to the projection of y_0 on the intersection of A and B.

We illustrate the method in Figure 2. We again successively project on both sets but in addition to what we do in the alternating projection method, we also remember the direction vector of projection denoted p and q. Except for the first two iterations, where the algorithm is the same as the alternating projection method, we add the direction vectors to the last point before the point is projected on the other set. If any of sets A and B is linear, the corresponding direction vector is parallel to the direction of projection and the intermediate step is not needed. If both sets are linear, Dykstra's algorithm reduces to the alternating projection method, using mathematical induction and

$$
\begin{aligned}
x_{k+1} &= P_A(y_k + p_k) = P_A(y_k), &(7)\\
y_{k+1} &= P_B(x_{k+1} + q_k) = P_B(x_{k+1}), &(8)
\end{aligned}
$$

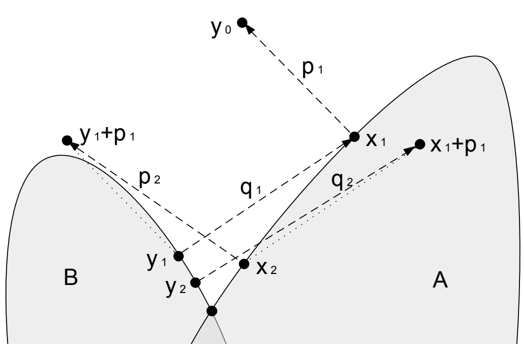

Figure 2. *Illustration of the Dykstra's projection algorithm.*

Alternating direction method of multipliers

The alternating direction method of multipliers (ADMM) [16], a variant of the augmented Lagrangian method [20, 6] with partial updates, is a popular optimization tool to minimize the sum of two functions

$$\min_x f(x) + g(Gx), \tag{9}$$

where functions f and g are convex not necessarily differentiable and G a linear operator. It consists of iteratively executing three update steps

$$
\begin{aligned}
x &\leftarrow \arg\min_x f(x) + \frac{\mu}{2}\|Gx - y - d\|^2, &(10)\\
y &\leftarrow \arg\min_y g(y) + \frac{\mu}{2}\|Gx - y - d\|^2, &(11)\\
d &\leftarrow d - (Gx - y), &(12)
\end{aligned}
$$

where scalar $\mu > 0$ is a parameter, y is an auxiliary variable (for sparsity applications representing a sparse domain counterpart of x) and d a dual variable. For $G = I$, both x and y converge to the minimum of (9). As a stopping criterion, we can use the distance of Gx and y, see discussion in [7].

ADMM for the feasibility problem

ADMM can be adapted to solve the feasibility problem considering indicator functions of the convex sets A and B. *The indicator function* of a subset C in a space X is a function $I_C : X \to \{0,1\}$ defined as

$$I_C(x) = \begin{cases} 1 \; if\, x \in C, \\ 0 \; if\, x \notin C, \end{cases} \tag{13}$$

i.e. the value of the indicator function is 1 for all elements of C and 0 for all elements of X not in C.

In our case, G is identity and f and g in (9) are indicator functions of A and B, giving iterations

$$
\begin{aligned}
x_{k+1} &= P_A(y_k - d_k), &(14)\\
y_{k+1} &= P_B(x_{k+1} + d_k), &(15)\\
d_{k+1} &= d_k + x_{k+1} - y_{k+1}, &(16)
\end{aligned}
$$

where $y_0 \in X$, $d_0 := 0$ and $k \in \mathbb{N}$.

The method is illustrated in Figure 3. Because we start with zero direction vector d_0, the first two iterations are again the same as in the alternating projection method or Dykstra's method. After getting y_1, you remember the direction vector $d_1 = x_1 - y_1$, which you subtract or add to a point before projecting it on A and B, respectively. In the next step you add the vector $x_2 - y_2$ to d_1 and continue analogically. In our example, the method converged in two iterations. In practice, iterations must be stopped using a suitable convergence criterion - either the difference between x and y, or checking if x is in the intersection. The method finds an arbitrary point in the intersection, not necessarily closest to the starting point.

ADMM for the best approximation problem

ADMM can be used to solve the best approximation problem, as well. Let I_A and I_B be indicator functions of A and B, *respectively.* Let us solve the best approximation problem as

$$\min_x \frac{1}{2}||x - z||^2 + I_A(x) + I_B(x). \tag{17}$$

Denoting $f(x) = \frac{1}{2}||x - z||^2 + I_A(x)$ and $g(x) = I_B(x)$ gives the ADMM as

$$
\begin{aligned}
x &\leftarrow \arg\min_x I_A(x) + \frac{1}{2}||x - z||^2 + \frac{\mu}{2}\|x - y - d\|^2, &(18)\\
y &\leftarrow P_B(x - d), &(19)\\
d &\leftarrow d - (x - y). &(20)
\end{aligned}
$$

In the first line

$$\frac{1}{2}||x-z||^2+\frac{\mu}{2}\|x-y-d\|^2= \tag{21}$$

$$=\frac{1+\mu}{2}x^Tx-x^T\left(z+\mu\left(y+d\right)\right) \tag{22}$$

$$+\frac{1}{2}\|z\|^2+\frac{\mu}{2}\|y+d\|^2\left(z+\mu\left(y+d\right)\right) \tag{23}$$

$$=\frac{1+\mu}{2}\left\|x-\frac{z+\mu\left(y+d\right)}{1+\mu}\right\|^2 \tag{24}$$

$$+\frac{1}{2}\|z\|^2+\frac{\mu}{2}\|y+d\|^2. \tag{25}$$

It means that the first line is still a projection and the algorithm becomes

$$x \leftarrow P_A\left(\frac{1}{1+\mu}z+\frac{\mu}{1+\mu}\left(y+d\right)\right), \tag{26}$$

$$y \leftarrow P_B\left(x-d\right), \tag{27}$$

$$d \leftarrow d-\left(x-y\right). \tag{28}$$

Figure 4 illustrates the modified method of ADMM for $\mu=\frac{1}{2}$.

For another modification, making ADMM approach more symmetric, we can consider $f(x)=\frac{1}{2}||x-z||^2+I_A(x)$ and $g(x)=\frac{1}{2}||x-z||^2+I_B(x)$ (rigorously $\frac{1}{4}$ but because of the indicator functions, it does not matter). We get

$$y \leftarrow \arg\min_y I_B(y)+\frac{1}{2}||y-z||^2+\frac{\mu}{2}\|x-y-d\|^2, \tag{29}$$

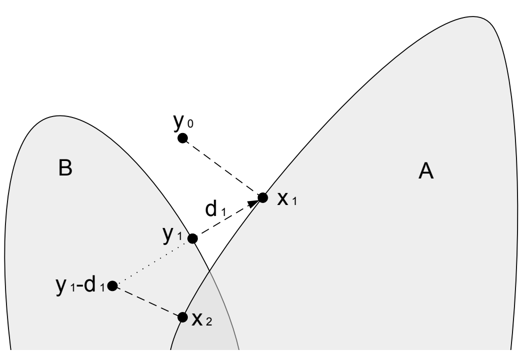

Figure 3. *Illustration of the alternating direction method of multipliers.*

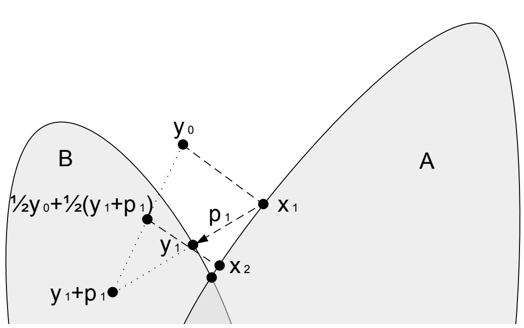

Figure 4. *Illustration of the ADMM for the best approximation problem.*

which is analogous to the first case (18) and the algorithm becomes

$$x \leftarrow P_A\left(\frac{1}{1+\mu}z+\frac{\mu}{1+\mu}\left(y+d\right)\right) \tag{30}$$

$$y \leftarrow P_B\left(\frac{1}{1+\mu}z+\frac{\mu}{1+\mu}\left(x-d\right)\right), \tag{31}$$

$$d \leftarrow d-\left(x-y\right). \tag{32}$$

This can be simplified by substitution $x'=\frac{\mu}{1+\mu}x$, analogously for y, d and $z'=\frac{1}{1+\mu}z$, giving

$$x' \leftarrow \frac{\mu}{1+\mu}P_A\left(z'+y'+d'\right)$$

$$y' \leftarrow \frac{\mu}{1+\mu}P_B\left(z'+x'-d'\right),$$

$$d' \leftarrow d'-\left(x'-y'\right).$$

and finally recovering $x=\frac{1+\mu}{\mu}x'$.

Application in Image Processing

In this section, we show an application of the projection methods on a simplified image processing problem. Let us assume we have two JPEG images of a slightly shifted scene. The goal is to improve the image quality using the projection methods.

JPEG compression

To understand the incorporation of the projection methods in the algorithm, we first shortly describe the principles of JPEG compression. JPEG uses a lossy form of compression based on the quantization of the discrete cosine transform (DCT). The basic workflow to create a JPEG file is as follows. First, colour images are transformed into YC_BC_R colour space and then each channel is handled separately. The next step is to downsample the chrominance channels C_B and C_R. DCT is then performed on blocks of usually 8×8 pixels. Finally, this converted matrix is quantized by a quantization table where we minimize the higher frequencies over the lower frequencies. This stage is the main lossy part of the algorithm.

The lossy part of JPEG compression can be expressed as

$$y=\left[QCDx\right], \tag{33}$$

where x is the vectorized original image, y the vector of coefficients stored in the JPEG file, Q, C and D are matrices and the square brackets denote rounding to the nearest integer. D is a downsampling matrix which returns the average value for every non-overlapping window of usually 2×2 pixels. C is a block-diagonal matrix of the block DCT and Q is a diagonal matrix of coefficients from the quantization table stored in each JPEG file. We denote the coefficient vector as q, thus $Q=diag\left(1/q\right)$.

The fact that JPEG works on the 8×8 grid is the reason why we need the two images to be shifted by a number of pixels not divisible by 8, otherwise they would be identical. If the 8×8 grids in JPEG are not aligned, we get different JPEG files which provide complementary information that can be used to obtain an image of higher quality.

Denoting the coefficients in input JPEG images as vectors y_1 and y_2, equation (33) implies that the image x should satisfy

$$QCDx \in \left\langle y_1 - \frac{1}{2}, y_1 + \frac{1}{2} \right\rangle, \quad (34)$$

$$QCDSx \in \left\langle y_2 - \frac{1}{2}, y_2 + \frac{1}{2} \right\rangle, \quad (35)$$

where S is the operator of shift between input images. The intervals in (34) and (35) are multi-dimensional intervals corresponding to the rounding in (33). Equations (34) and (35) specify two convex sets, the original image x should belong to. Therefore, the aim is to find an image x in the intersection of the sets using the projection methods described in the previous sections.

Experiments

In this section, we demonstrate convergence properties of the projection methods applied on the image processing problem described above. We used a set of 59 images of resolution 433×650 pixels. We crop the images by taking 392×608 pixels from the upper-left and bottom-right corners to simulate the shift. The images are saved as JPEG files using Matlab with the quality setting of 80. The first of the images is then used as the initial image in the projection algorithms.

All projection algorithms require the sets that we project on to be closed. Therefore, instead of intervals (34) and (35) we work with intervals $\left\langle y - \frac{1}{2}, y + \frac{1}{2} - \delta \right\rangle$ where δ is sufficiently small. We also need the projections onto the intervals. It can be shown that the projections can be expressed as

$$P_{QCDx \in \left\langle y-\frac{1}{2}, y+\frac{1}{2}-\delta \right\rangle}(z) = z - \frac{1}{k} D^T C^T diag(q) \cdot \quad (36)$$

$$\cdot \left(QCDz - P_{QCDx \in \left\langle y-\frac{1}{2}, y+\frac{1}{2}-\delta \right\rangle}(QCDz) \right), \quad (37)$$

where constant k is a downsampling factor ($1/4$ for 2×2 downsampling) [19].

The quality of the resulting images improved on average from $37.47db$ to $38.37db$. This is not a big difference and in practice, the algorithm would have to incorporate stronger assumptions such as sparse priors. The purpose of this experiment was to illustrate the difference in convergence between available algorithms, though.

Figure 5 shows the speed of convergence for each method. The value on y-axis is the mean error, i. e. the mean distance from the point of convergence. As expected, Dykstra algorithm and the ADMM for the best approximation problem follow a similar curve, converging after about 1100 iterations. These two methods are significantly slower than POCS, with about 380 iterations to converge, which is in turn much slower than the ADMM for the feasibility problem, which converged on average in 12 iterations.

Conclusion

In this paper, we compare efficiency of the classical algorithm for computing points in the intersection of two sets, POCS and Dykstra's algorithms with the algorithms based on the ADMM. The ADMM algorithm for the feasibility problem is much faster than POCS, which is a well known fact from literature. The Dykstra's algorithm is comparable with the ADMM for the best approximation problem. In practice, it is important to distinguish between the feasibility and best approximation problems and, if possible, to use much faster algorithms for the feasibility problem. Even the simple POCS algorithm applied on the feasibility problem is much faster than all best approximation algorithms.

Acknowledgments

The work of Zuzana Bílková was funded by the GAUK, grant No.1583117 and the grant SVV– 2017–260452, the work of Michal Šorel by the Grant Agency of the Czech Republic under project GA18-05360S.

References

[1] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo, *An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems*, IEEE Transactions on Image Processing, 20 (2010), pp. 681–695.

[2] R. G. Baraniuk, *Compressive sensing [lecture notes]*, IEEE signal processing magazine, 24 (2007), pp. 118–121.

[3] H. H. Bauschke and J. M. Borwein, *On projection algorithms for solving convex feasibility problems*, SIAM review, 38 (1996), pp. 367–426.

[4] H. H. Bauschke and V. R. Koch, *Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces*, Contemporary Mathematics, 636 (2015), pp. 1–40.

[5] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM journal on imaging sciences, 2 (2009), pp. 183–202.

[6] D. P. Bertsekas, *Nonlinear programming (2nd ed.)*, Athena scientific, Belmont, MA, 1999.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine Learning, 3 (2011), pp. 1–122.

[8] L. M. Briceno-Arias and P. L. Combettes, *Monotone operator methods for nash equilibria in non-potential games*, in Computational and Analytical Mathematics,

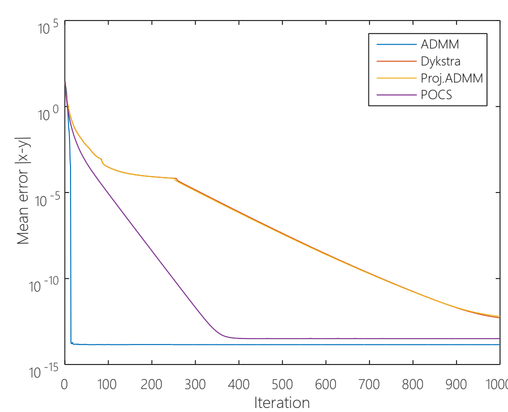

Figure 5. *Speed of convergence of projection methods.*

Springer, 2013, pp. 143–159.
[9] Y. Censor, W. Chen, P. L. Combettes, R. Davidi, and G. T. Herman, *On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints*, Computational Optimization and Applications, 51 (2012), pp. 1065–1088.
[10] A. Chambolle and T. Pock, *A first-order primal-dual algorithm for convex problems with applications to imaging*, Journal of mathematical imaging and vision, 40 (2011), pp. 120–145.
[11] W. Cheney and A. A. Goldstein, *Proximity maps for convex sets*, Proceedings of the American Mathematical Society, 10 (1959), pp. 448–450.
[12] P. L. Combettes and J.-C. Pesquet, *Proximal splitting methods in signal processing*, in Fixed-point algorithms for inverse problems in science and engineering, Springer, 2011, pp. 185–212.
[13] L. Condat, *A generic proximal algorithm for convex optimization - application to total variation minimization*, IEEE Signal Processing Letters, 21 (2014), pp. 985–989.
[14] F. R. Deutsch, *Best approximation in inner product spaces*, Springer Science & Business Media, 2012.
[15] R. L. Dykstra, *An algorithm for restricted least squares regression*, Journal of the American Statistical Association, 78 (1983), pp. 837–842.
[16] J. Eckstein and D. P. Bertsekas, *On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators*, Mathematical Programming, 55 (1992), pp. 293–318.
[17] M. Fukushima, *The primal douglas-rachford splitting algorithm for a class of monotone mappings with application to the traffic equilibrium problem*, Mathematical Programming, 72 (1996), pp. 1–15.
[18] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*, vol. 9, SIAM, 1989.
[19] A. Novozámskỳ and M. Šorel, *Detection of copy-move image modification using jpeg compression model*, Forensic science international, 283 (2018), pp. 47–57.
[20] S. Wright and J. Nocedal, *Numerical optimization*, Springer Science, 1999.

Author Biography

Zuzana Bílková is a PhD. student at Charles University in Prague and she works in the Czech Academy of Sciences. She is oriented on applications of deep neural networks in image processing. She is a holder of two grants, one from Charles University on convolutional neural networks and the second from the Technology Agency of the Czech Republic.