An alternating projection method with memory

Janosch Rieger

May 2, 2019

Abstract

We propose a variant of the method of alternating projections, which uses the lengths of projection steps carried out in the past to decide which projections to choose in the future. We prove convergence of this algorithm and demonstrate that it outperforms the method of alternating projections in some numerical examples.

MSC Codes: 90C25, 90C59

Keywords: Method of alternating projections, acceleration of convergence, feasibility problem

1 Introduction

The method of alternating projections is a well-established numerical algorithm for computing a point in the intersection of finitely many closed convex subsets of a Hilbert space, see [1], [5], [8] and the references therein. In principle, the original algorithm can be arbitrarily slow, see [7] for a pathological example, and it has been observed to be slow when applied to real-world problems. For this reason, accelerated variants of the original scheme, often based on line-search ideas, have been proposed, see e.g. [2] and [9].

The guiding idea behind the numerical method presented in this paper is to gather as much information on the relative geometry of the closed convex sets as possible without significant additional computational effort, and to use this information to achieve an acceleration of the method of alternating
projections. An inspection of the convergence proof of the original method in [3] reveals that the performance of the algorithm is completely determined by the lengths of the projection steps carried out. As these lengths are easy to compute, our algorithm stores the lengths of projection steps carried out in the past and uses them to decide which projections to choose in the future.

We prove that this method converges, and we demonstrate that our approach indeed outperforms the method of alternating projections in some numerical examples. As to be expected, this seems to happen in particular when the problem at hand is ill-conditioned, i.e. angles between several of the sets are small.

2 The algorithm

Given closed convex sets $C_1, \ldots, C_N \subset \mathbb{R}^d$ with $C := \cap_{j=1}^{N} C_j \neq \emptyset$, we wish to find a point $x^* \in C$. The method of alternating projections, see [3], achieves this by carrying out the iteration

$$x_{k+1} := \text{proj}(x_k, C_{\text{mod}(k,N)+1}) \quad \forall k \in \mathbb{N}.$$

This sequence may converge very slowly when many of the projection steps $\text{proj}(x_k, C_{\text{mod}(k,N)+1}) - x_k$ are small. The idea behind Algorithm 1 is to keep a record of the lengths of projection steps performed in the past and to give preference to operations that have lead to large projection steps. Our approach resembles to some extent the techniques of loping and flagging introduced in [6] to suppress noise in the data, which is achieved by ignoring updates with very small residuals in the method of alternating projections. Its behavior is, however, quite different.

The sequence of matrices $(D_k)_{k \in \mathbb{N}}$ records more or less the length of the last projection step from set $C_i$ to $C_j$ in its $(i,j)$-th component, and the sequence $(\alpha_k)_{k \in \mathbb{N}}$ keeps track of the minimal entry in $D^k$. The construction of the sequence $(\delta_k)_{k \in \mathbb{N}}$ ensures that every entry of $D^k$ is strictly positive. The parameter $\beta$ decides how close to zero an entry can be relative to the rest of the matrix, or, in other words, at which performance level of the overall scheme to reconsider an operation that has performed very badly in the past. These precautions seem to be necessary from a phenomenological point of view, and they are essential for the proof of Proposition 3.

When $N > 100$, it is not economical to store the past information in a matrix, because searching for the maximal element of a row becomes too
Algorithm 1: An alternating projections algorithm with memory

**Input:** sets $C_1, \ldots, C_N \subset \mathbb{R}^d$, $x \in \mathbb{R}^d$, $\alpha_0 > 0$, $\beta \in (0, 1)$

1. $D^0 \leftarrow \alpha_0 \ast \text{ones}(N, N)$
2. $x_0 \leftarrow \text{proj}(x, C_1)$
3. $j_0 \leftarrow 1$
4. for $k \leftarrow 0$ to $\infty$ do
   5. /* carry out most promising projection */
   6. $j_{k+1} \leftarrow \text{uniform_random_sample}(\arg \max_{\ell \in \{1, \ldots, N\} \setminus \{j_k\}} D^k_{j_k, \ell})$
   7. $x_{k+1} \leftarrow \text{proj}(x_k, C_{j_{k+1}})$
   8. /* update distance matrix and lower bound */
   9. $\delta_k \leftarrow \max \{\|x_{k+1} - x_k\|, \beta \alpha_k\}$
   10. $D^{k+1} \leftarrow D^k$
   11. $D^{k+1}_{j_k, j_{k+1}} \leftarrow \delta_k$
   12. $\alpha_{k+1} \leftarrow \min \{\alpha_k, \delta_k\}$
5. end

expensive. We recommend using one self-balancing binary search tree per row of $D$ to store this information.

3 Convergence analysis

We restate a slightly modified version of Corollaries 1 and 2 from [3].

**Lemma 1.** Let $C_1, \ldots, C_N \subset \mathbb{R}^d$ be closed convex sets with $\cap_{j=1}^N C_j \neq \emptyset$, and let the sequences $(j_k)_{k \in \mathbb{N}} \in \{1, \ldots, N\}^\mathbb{N}$ and $(x_k)_{k \in \mathbb{N}} \in (\mathbb{R}^d)^\mathbb{N}$ satisfy

$$x_{k+1} = \text{proj}(x_k, C_{j_k}) \quad \forall k \in \mathbb{N}.$$ 

Then we have

$$\|x_{k+1} - z\|^2 \leq \|x_k - z\|^2 - \|x_{k+1} - x_k\|^2 \quad \forall k \in \mathbb{N}, \quad (1)$$

$$\|x_{k+1} - z\| \leq \|x_k - z\| \leq \|x_0 - z\| \quad \forall k \in \mathbb{N}, \quad (2)$$

In principle, Proposition 2 can be considered a corollary to Theorem 4.1 from [4]. We include our own proof for two reasons. On one hand, we would like to keep the paper self-contained and accessible for readers who do not wish to familiarize themselves with the theory of nonexpansive operators.
On the other hand, our proof is purely based on an assessment of the lengths of projection steps, and it thus shares a common theme with Algorithm 1.

**Proposition 2.** Let $C_1, \ldots, C_N \subset \mathbb{R}^d$ be closed convex sets with $\cap_{j=1}^N C_j \neq \emptyset$, and let $(j_k)_{k \in \mathbb{N}} \in \{1, \ldots, N\}^\mathbb{N}$ and $(x_k)_{k \in \mathbb{N}} \in (\mathbb{R}^d)^\mathbb{N}$ be sequences which satisfy

$$x_{k+1} = \text{proj}(x_k, C_{j_k}), \quad \forall k \in \mathbb{N}$$

as well as the recurrence condition

$$\# \{k \in \mathbb{N} : j_k = j \} = \infty \quad \forall j \in \{1, \ldots, N\}.$$  \hspace{1cm} (3)

Then there exists $x^* \in \cap_{j=1}^N C_j$ such that $\lim_{k \to \infty} x_k = x^*$.

**Proof.** Because of statement (2) of Lemma 1, there exist a subsequence $(x_{k\ell})_{\ell \in \mathbb{N}}$ of $(x_k)_{k \in \mathbb{N}}$ and $x^* \in \mathbb{R}^d$ such that

$$\lim_{\ell \to \infty} \|x_{k\ell} - x^*\| = 0.$$  \hspace{1cm} (4)

Clearly, there exist $j^* \in \{1, \ldots, N\}$ and a subsequence $(k_{\ell m})_{m \in \mathbb{N}}$ of the sequence $(k_\ell)_{\ell \in \mathbb{N}}$ with

$$j_{k_{\ell m}} = j^* \quad \forall m \in \mathbb{N}.$$  

Since $C_{j^*}$ is closed, we have $x^* \in C_{j^*}$. We partition $\{1, \ldots, N\}$ into

$$J^* := \{ j \in \{1, \ldots, N\} : x^* \in C_j \}, \quad J_* = \{1, \ldots, N\} \setminus J^*.$$  

By the above, we have $J^* \neq \emptyset$. Assume that $J_* \neq \emptyset$. By induction, using statement (3), we can construct sequences $(k'_\ell)_{\ell \in \mathbb{N}} \in \mathbb{N}^\mathbb{N}$ and $(k''_\ell)_{\ell \in \mathbb{N}} \in \mathbb{N}^\mathbb{N}$ given by

$$k'_0 := k_{t_0}, \quad k''_0 := \min \{ k : k > k_{t_0}, j_k \in J_* \},$$

$$k'_{\ell+1} := \min \{ k_{\ell m} : k_{\ell m} > k''_\ell \},$$

$$k''_{\ell+1} := \min \{ k : k > k'_{\ell+1}, j_k \in J_* \},$$

so that, in particular, we have

$$k'_0 < k''_0 < k'_1 < k''_1 < \ldots$$

Now there exists $\varepsilon > 0$ such that

$$\text{dist}(x^*, C_j) \geq 2\varepsilon \quad \forall j \in J_*.$$
By construction of the sequence \((k'_\ell)_{\ell \in \mathbb{N}}\), there exists \(L \in \mathbb{N}\) such that
\[
\|x_{k'_\ell} - x^*\| \leq \varepsilon \quad \forall \ell \geq L.
\]
Applying statement (2) of Lemma 1 with \(z = x^*\) and the system of sets \(\{C_j : j \in J^*\}\), and using the construction of the sequence \((k''_\ell)_{\ell \in \mathbb{N}}\), we obtain
\[
\|x_k - x^*\| \leq \|x_{k'_\ell} - x^*\| \leq \varepsilon \quad \forall k \in [k'_\ell, k''_\ell), \ \forall \ell \in \mathbb{N}.
\]
On the other hand, we have \(\|x_{k''_\ell} - x^*\| \geq 2\varepsilon\) for all \(\ell \in \mathbb{N}\), so
\[
\|x_{k''_\ell} - x_{k''_{\ell-1}}\| \geq \|x_{k''_\ell} - x^*\| - \|x^* - x_{k''_{\ell-1}}\| \geq \varepsilon \quad \forall \ell \in \mathbb{N}. \quad (5)
\]
Now let \(z \in \bigcap_{j=1}^{N} C_j\) and use statement (5) and statement (1) from Lemma 1 multiple times to obtain
\[
\lim_{k \to \infty} \|x_k - z\|^2 \leq \|x_0 - z\|^2 - \sum_{j=0}^{k-1} \|x_{j+1} - x_j\|^2 = -\infty,
\]
which is a contradiction. Hence \(J^* = \emptyset\), and \(x^* \in \bigcap_{j=1}^{N} C_j\). Now statements (4) and statement (2) of Lemma 1 with \(z = x^*\) imply \(\lim_{k \to \infty} x_k = x^*\), as desired.

We check that Algorithm 1 satisfies the assumptions of Proposition 2.

**Proposition 3.** Let \(C_1, \ldots, C_N \subset \mathbb{R}^d\) be closed convex sets with \(\bigcap_{j=1}^{N} C_j \neq \emptyset\), let \(x_0 \in \mathbb{R}^d\), let \(\alpha_0 > 0\), and let \(\beta \in (0,1)\). Then the sequences \((j_k)_{k \in \mathbb{N}} \in \{1, \ldots, N\}^\mathbb{N}\), \((\alpha_k)_{k \in \mathbb{N}} \in \mathbb{R}_0^\mathbb{N}\), \((\delta_k)_{k \in \mathbb{N}} \in \mathbb{R}_0^\mathbb{N}\), \((D^k)_{k \in \mathbb{N}} \in (\mathbb{R}^{N \times N})^\mathbb{N}\) and \((x_k)_{k \in \mathbb{N}} \in (\mathbb{R}^d)^\mathbb{N}\) generated by Algorithm 1 satisfy
\[
\lim_{k \to \infty} \alpha_k = 0, \quad \lim_{k \to \infty} \delta_k = 0, \quad \lim_{k \to \infty, m \neq n} \max_{k \to \infty, m \neq n} D^k_{m,n} = 0,
\]
and we have
\[
\#\{k \in \mathbb{N} : j_k = j\} = \infty \quad \forall j \in \{1, \ldots, N\}.
\]

**Proof.** By lines 1, 5 and 8 of Algorithm 1, the sequences \((\alpha_k)_{k \in \mathbb{N}}\), \((\delta_k)_{k \in \mathbb{N}}\) and \((D^k_{m,n})_{k \in \mathbb{N}}\) with \(m, n \in \{1, \ldots, N\}\) have strictly positive elements. Let \(z \in \bigcap_{j=1}^{N} C_j\). Applying inequality (1) from Lemma 1 multiple times yields
\[
\|x_k - z\|^2 \leq \|x_0 - z\|^2 - \sum_{j=0}^{k-1} \|x_{j+1} - x_j\|^2 \quad \forall k \in \mathbb{N},
\]
which forces
\[
\lim_{k \to \infty} \| x_{k+1} - x_k \| = 0.
\] (6)

It follows from (6) and the recursion
\[
\alpha_{k+1} = \min\{\alpha_k, \delta_k\} \leq \delta_k = \max\{\| x_{k+1} - x_k \|, \beta \alpha_k\}
\]
that \(\lim_{k \to \infty} \alpha_k = 0\) and \(\lim_{k \to \infty} \delta_k = 0\). Now we define
\[
J := \{1, \ldots, N\} \setminus \{(m, m) : m \in \{1, \ldots, N\}\},
\]
\[
J_\infty := \{(m, n) \in J : \#\{k \in \mathbb{N} : (m, n) = (j_k, j_{k+1})\} = \infty\},
\]
\[
J_0 := J \setminus J_\infty.
\]

Since \(\#J < \infty\), there exists \((m^*, n^*) \in J_\infty\), and since \(\lim_{k \to \infty} \delta_k = 0\), we have
\[
\lim_{k \to \infty} \max_{(m, n) \in J_\infty} D_{m,n}^k = 0.
\] (7)

On the other hand, there exists \(\varepsilon > 0\) with
\[
D_{m,n}^k \geq \varepsilon \quad \forall (m, n) \in J_0, \forall k \in \mathbb{N}.
\] (8)

By line 3 of the algorithm and statement (7), we have
\[
\liminf_{k \to \infty} D_{m^*,n}^k \leq \liminf_{k \to \infty} D_{m^*,n^*}^k = 0 \quad \forall n \in \{1, \ldots, N\} \setminus \{m^*\},
\]
and hence, using statement (8), we obtain
\[
(m^*, n) \in J_\infty \quad \forall n \in \{1, \ldots, N\} \setminus \{m^*\}.
\]

As a consequence, we have
\[
\#\{k \in \mathbb{N} : j_k = m\} = \infty \quad \forall m \in \{1, \ldots, N\}.
\]

Again, since \(\{1, \ldots, N\} < \infty\), this implies that
\[
\forall m \in \{1, \ldots, N\} \exists n_m \in \{1, \ldots, N\} \setminus \{m\} : (m, n_m) \in J_\infty.
\]

By the same argument as above, applied to \((m, n_m)\) for all \(m \in \{1, \ldots, N\}\) instead of \((m^*, n^*)\), we obtain that \(J_\infty = J\). All in all, we have proved the desired statement. \(\square\)
Now we summarize the above in the main result of this paper.

**Theorem 4.** Let $C_1, \ldots, C_N \subset \mathbb{R}^d$ be closed convex sets with $\cap_{j=1}^N C_j \neq \emptyset$, let $x_0 \in \mathbb{R}^d$, let $\alpha_0 > 0$, and let $\beta \in (0, 1)$. Then there exists $x^* \in \cap_{j=1}^N C_j$ such that the sequence $(x_k)_{k \in \mathbb{N}} \in (\mathbb{R}^d)^\mathbb{N}$ generated by Algorithm 1 satisfies

$$\lim_{k \to \infty} x_k = x^*.$$

**Proof.** Proposition 3 verifies the assumptions of Proposition 2 in the situation of Algorithm 1, which guarantees convergence of the sequence $(x_k)_{k \in \mathbb{N}}$. □

### 4 Numerical examples

We compare our algorithm (APMem) not only with the method of alternating projections (MAP), but also with the method of shuffled alternating projections (SAP), which randomly shuffles the order of the projections $\text{proj}(\cdot, C_j)$ in every cycle. This allows us to distinguish acceleration achieved by using past information from other effects related to a specific labeling of the subspaces.

For large $\alpha_0$, our method APMem behaves like SAP in an initial learning phase, while for small $\alpha_0$, it behaves like MAP initially. Since SAP usually outperforms MAP, we recommend choosing a fairly high $\alpha_0$. The choice of $\beta$ does not seem to impact the performance of the algorithm too much, which is probably due to the fact that we test the method on rather simple sets. For this reason, we kept $\beta = 1/100$ throughout the numerical experiments.

The first example is deliberately chosen in such a way that alternating projection type methods struggle as much as possible.

**Example 5.** We consider the almost identical one-dimensional subspaces

$$C_j := \{ s \left( \cos\left(\frac{j-1}{N} \gamma \right), \sin\left(\frac{j-1}{N} \gamma \right) \right) : s \in \mathbb{R} \} \quad \forall j \in \{1, \ldots, N - 1\}$$

and the outlier

$$C_N := \{ s \left( \cos(10\gamma), \sin(10\gamma) \right) : s \in \mathbb{R} \}.$$

For aesthetical reasons, we choose $N = 10$, $\gamma = \pi/200$ and the initial point $x_0 = (1, 0)$. 7
Figure 1: Iterates (red) of the numerical algorithms applied to subspaces (blue) from Example 5 with $N = 10$. APMem with $\alpha_0 = 1$ and $\beta = 0.01$.

Figure 2: Frequencies (yellow=high, blue=low) of transitions from set $C_i$ to set $C_j$ in Example 5 with $N = 10$. APMem with $\alpha_0 = 1$ and $\beta = 0.01$.

Figure 3: Error plots for Example 5. MAP depicted in blue, SAP in red, APMem (with $\alpha_0 = 1$ and $\beta = 0.01$) in yellow.
Figure 1 shows the first 40 cycles or 400 iterates of every scheme. Initially, all methods behave in a similar way, but after some time, APMem oscillates more frequently between the outlier $C_{10}$ and the other subspaces than the other two methods, which implies that it covers longer distances on average and hence makes more progress.

Figure 2 shows the frequencies with which the numerical methods transit from one set $C_i$ to another $C_j$. While MAP deterministically moves from one set to the next, SAP migrates according to a uniform random distribution. Our algorithm APMem, however, almost exclusively alternates between the outlier $C_{10}$ and some arbitrary other subspace, which is exactly the behavior we wish for.

Figure 3 compares the errors of MAP, SAP and APMem for different numbers of subspaces, leaving all other parameters fixed. In this example, MAP and SAP exhibit an almost identical performance, while APMem clearly outperforms them after an initial learning phase. Again, this is exactly the behavior we expect.

Example 6. We consider the following model problem. For given radius $r > 0$ and angle $\gamma \in (0, 2\pi]$, we define the one-dimensional subspaces

$$C_j := \{ s \begin{pmatrix} r \cos(j \frac{\pi}{N} \gamma) \\ r \sin(j \frac{\pi}{N} \gamma) \\ 1 \end{pmatrix} : s \in \mathbb{R} \} \quad \forall j \in \{1, \ldots, N\}.$$ 

For aesthetical reasons, we choose parameters $N = 9$ and $\gamma = \pi$, and the initial point $x_0 = (\cos(\frac{\pi}{9}), \sin(\frac{\pi}{9}), 1)$.

Figure 4 shows the first 36 cycles or 324 iterates of every scheme. Apparently, the orderly fashion in which MAP proceeds is less helpful for making progress than the random exploration of the geometry by SAP. The iterates of APMem cluster at the two subspaces with the largest angle, which leads to an acceleration. These behaviors are reflected by the transit frequencies shown in Figure 5.

Figure 6 compares the errors of MAP, SAP and APMem for different numbers of subspaces and different degrees of ill-conditioning, incorporated by the parameter $r > 0$, leaving all other parameters fixed. Obviously, APMem always outperforms SAP after an initial learning phase, and SAP always outperforms MAP. Apart from that, two trends can be clearly recognized from this figure. The smaller $r > 0$, i.e. the more ill-conditioned the problem.
becomes, the more distinct becomes the difference in performance between APMem and SAP. On the other hand, the larger the number of subspaces \( N \) becomes, the smaller the difference in performance between APMem and SAP.

Figure 7 reveals, to a certain extent, the reason for this behavior. If the subspaces are densely packed, relative to the size of the largest gap between \( C_1 \) and \( C_N \), then the lengths of the projection steps approximate a continuum, which leads to insufficient contrast between step-lengths observed in the past. For that reason, the APMem scheme carries out more and more low quality transitions in an almost random fashion, so that eventually it becomes indistinguishable from SAP.

We were not able to record the frequency matrix for the case \( N = 625 \) and \( r = 0.1 \), because in this case, the precision of the iterates at the end of the learning phase is close to the precision of the arithmetic system used. For this reason, we replaced this image by a placeholder.
Figure 6: Error plots for Example [6]. MAP depicted in blue, SAP in red, AP Mem (with $\alpha_0 = 1$ and $\beta = 0.01$) in yellow.

Figure 7: Transition frequencies of AP Mem for Example [6]. The top right picture cannot be computed, because too many iterates are indistinguishable from exact solution up to working precision, compare Figure [6].
Figure 8: Same as Figure 6 but with $\gamma = 2\pi$.

Figure 9: Same as Figure 7 but with $\gamma = 2\pi$. 
Figures 8 and 9 depict the same algorithms applied to the same problem as in Figures 6 and 7, but with parameter $\gamma = 2\pi$. They confirm the trends mentioned above and support the intuition behind APMem in the sense that the two bright bands in most of the matrices shown in Figure 9 correspond to transitions between antipodal subspaces.

5 Conclusion

We presented and analyzed a numerical method that seems to outperform other methods of alternating projection type on small and medium sized problems. As demonstrated in the computational examples, our method is particularly useful when applied to ill-conditioned problems, and when the angles between the sets do not approximate a continuum, but are to some extent clustered.

For large-scale problems, the learning phase of our method is too long, and it becomes a challenge to store the required information about past iterations. For this reason, it would be desirable to develop sparse versions of our algorithm, which could, for example, be based on multilevel techniques.

Acknowledgement

The author thanks Matthew Tam for an introduction to the world of projection methods and support during the preparation of this paper.

References

[1] H.H. Bauschke and P.L. Combettes. Convex analysis and monotone operator theory in Hilbert spaces. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011.

[2] H.H. Bauschke, F. Deutsch, H. Hundal, and S.-H. Park. Accelerating the convergence of the method of alternating projections. Trans. Amer. Math. Soc., 355(9):3433–3461, 2003.

[3] L.M. Brègman. Finding the common point of convex sets by the method of successive projection. Dokl. Akad. Nauk SSSR, 162:487–490, 1965.
[4] M.N. Dao and M.K. Tam. Union Averaged Operators with Applications to Proximal Algorithms for Min-Convex Functions. *J. Optim. Theory Appl.*, 181(1):61–94, 2019.

[5] F. Deutsch. *Best approximation in inner product spaces*, volume 7 of *CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC*. Springer-Verlag, New York, 2001.

[6] T. Elfving, P.C. Hansen, and T. Nikazad. Convergence analysis for column-action methods in image reconstruction. *Numer. Algorithms*, 74(3):905–924, 2017.

[7] C. Franchetti and W. Light. On the von Neumann alternating algorithm in Hilbert space. *J. Math. Anal. Appl.*, 114(2):305–314, 1986.

[8] A. Galántai. *Projectors and projection methods*, volume 6 of *Advances in Mathematics (Dordrecht)*. Kluwer Academic Publishers, Boston, MA, 2004.

[9] W.B. Gearhart and M. Koshy. Acceleration schemes for the method of alternating projections. *J. Comput. Appl. Math.*, 26(3):235–249, 1989.