Project Report for Seminar Course in Numerical Analysis, VT23

Junzi Zhang, Brendan O'Donoghue, Stephen Boyd: Globally Convergent Type-I Anderson Acceleration for Non-Smooth Fixed-Point Iterations

Theo Koppenhöfer

Lund

July 3, 2023

Input: Initial value $x_0 \in \mathbb{R}^n$ and function $f\colon \mathbb{R}^n \to \mathbb{R}^n$.

for $k = 0, 1, \dots$ **do**
  Set $x_{k+1} = f(x_k)$.
end

Algorithm 1: Fixed point iteration (original)

Introduction

The following report will summarise and present the main results of [1] as part of the course NUMN27, Seminar in Numerical Analysis. First we will motivate the Anderson-acceleration-I (AA-I) algorithm. Then we discuss the modifications made to the algorithm in [1]. After that we will give the main convergence result and finally test the AA-I algorithm in numerical experiments. The report, the Python implementation and the corresponding presentation of the topic can be found online under [2].

Motivation of the Anderson-Acceleration algorithm

In science it is a common problem to find a fixed point $x = f(x) \in \mathbb{R}^n$ of a function $f\colon \mathbb{R}^n \to \mathbb{R}^n$. An equivalent formulation is to find a zero $x \in \mathbb{R}^n$ of the function $g = \mathrm{Id} - f$. In the following we assume that f indeed has a fixed point and that f is non-expansive, i.e. we have for all $x, y \in \mathbb{R}^n$ that

$$\|f(x) - f(y)\| \leq \|x - y\|.$$

We however also assume that ∇f is unknown which means we cannot use Newton iteration to solve our problem. We assume that our problem is noisy so that we cannot take finite difference derivatives. If the cost of evaluating f is very high, then line search becomes unfeasible and if n is very large we are forced to work matrix free. We know that this type of problem can nonetheless be solved by the fixed point iteration described in algorithm 1.

It can be shown that the fixed point iteration will in this case converge to a fixed point of f. This convergence is slow if the Lipschitz-constant of f is close to 1. It is here were the Anderson-Acceleration (AA) methods come in.

Input: $x_0 \in \mathbb{R}^n$ and $f\colon \mathbb{R}^n \to \mathbb{R}^n$.

for $k = 0, 1, \dots$ **do**
  Set $f_k = f(x_k)$.
  Choose $\alpha = \alpha^k \in \mathbb{R}^{k+1}$ such that $\sum_i \alpha_i = 1$.
  Set $x_{k+1} = \sum_i \alpha_i f_i$.
end

Algorithm 2: General AA

Input: $x_0 \in \mathbb{R}^n$ and $f\colon \mathbb{R}^n \to \mathbb{R}^n$.

for $k = 0, 1, \dots$ **do**
  Set $f_k = f(x_k)$.
  Set $g_k = x_k - f_k$.
  Choose $\alpha \in \mathbb{R}^{k+1}$ such that $\sum_i \alpha_i = 1$ and such that α minimises $\|\sum_i \alpha_i g_i\|$.
  Set $x_{k+1} = \sum_i \alpha_i f_i$.
end

Algorithm 3: AA-II

Input : $x_0 \in \mathbb{R}^n$ and $f\colon \mathbb{R}^n \to \mathbb{R}^n$.

Set $x_1 = f(x_0)$.
for $k = 0, 1, \dots$ **do**
  Set $g_k = g(x_k)$.
  Construct $S_k = \begin{bmatrix} x_1 - x_0 & \cdots & x_k - x_{k-1} \end{bmatrix} \in \mathbb{R}^{n\times k}$ and $Y_k = \begin{bmatrix} g_1 - g_0 & \cdots & g_k - g_{k-1} \end{bmatrix} \in \mathbb{R}^{n\times k}$.
  Set $H_k = \mathrm{Id} + (S_k - Y_k)\left(Y_k^\top Y_k\right)^{-1} Y_k^\top \in \mathbb{R}^{n\times n}$.
  Set $x_{k+1} = x_k - H_k g_k$.
end

Algorithm 4: AA-II (reformulated)

The main idea of AA methods is to use the information gained from previous function evaluations of f to determine the point x_{k+1}. To update x_{k+1} we now form a weighted average as described in algorithm 2. For simplicity of notation we assume in the following that our memory is unlimited.

The General AA algorithm requires a specific choice of $\alpha \in \mathbb{R}^{k+1}$. Since finding a fixed point of f is equivalent to finding a zero of $g = \mathrm{Id} - f$ it seems sensible to require α to minimise

$$\left\| \sum_i \alpha_i g_i \right\|$$

under the condition that $\sum_i \alpha_i = 1$. This choice of α yields the AA-II algorithm given in 3.

It is shown in [1, Section 2.2] that this update can be brought into the form of a quasi-Newton-like method given in algorithm 4. This method bears a lot of similarities with the bad Broyden method where H_k is an approximate inverse of $\nabla f(x_k)$. Indeed one can show the following proposition.

Proposition 1 (Approximate inverse Jacobian)**.** *In algorithm 4 the matrix H_k minimises $\|H_k - \mathrm{Id}\|_F$ under the multi-secant condition $H_k S_k = Y_k$.*

Proof. See [1, Section 2.2]. □

The good Broyden method approximates the Jacobian rather than its inverse and tends to yield better results than the bad Broyden. This motivates to choose $B_k = H_k^{-1}$ to be a minimiser of $\|B_k - \mathrm{Id}\|_F$ under the condition $B_k Y_k = S_k$. If one chooses B_k in this way it is motivated in [1, Section 2.3] that

$$B_k = \mathrm{Id} + (Y_k - S_k)\left(S_k^\top S_k\right)^{-1} S_k^\top .$$

This yields the AA-I algorithm 5.

Input : $x_0 \in \mathbb{R}^n$ and $f\colon \mathbb{R}^n \to \mathbb{R}^n$.

Set $x_1 = f(x_0)$
for $k = 0, 1, \dots$ **do**
  Set $g_k = g(x_k)$.
  Construct S_k from $x_0, \dots, x_k$ and Y_k from $g_0, \dots, g_k$.
  Set $B_k = \mathrm{Id} + (Y_k - S_k)\left(S_k^\top S_k\right)^{-1} S_k^\top \in \mathbb{R}^{n\times n}$.
  Set $x_{k+1} = x_k - H_k g_k$ with $H_k = B_k^{-1}$.
end

Algorithm 5: AA-I

Modifications of the AA-I algorithm

The AA-I algorithm as stated in 5 has some issues. For one, the approach is not matrix-free. This is fixed by a rank-1 update formula for matrices B_k and later for H_k. It may also occur in a step that the matrix H_k is not well-defined. This may occur if B_k itself is not well-defined or is singular. The well-definedness will be resolved by a Powell-type regularisation and the restarting of the iteration. The restarting of the iteration will also yield an algorithm that does not require unlimited memory. Lastly, this algorithm does not always converge when the fixed point iteration algorithm converges. This will be resolved by adding a safeguarding of steps.

We start with the rank-1 update formula for B_k. One can show that

Proposition 2 (Rank-1 update for B_k)**.** *We have*

$$B_k = B_{k-1} + \frac{(y_{k-1} - B_{k-1}s_{k-1})\hat{s}_{k-1}^\top}{\hat{s}_{k-1}^\top s_{k-1}} \tag{1}$$

where $y_{k-1} = g_k - g_{k-1}$, $B_0 = \mathrm{Id}$ *and*

$$\hat{s}_{k-1} = s_{k-1} - \sum_{j=0}^{k-2} \frac{\hat{s}_j^\top s_{k-1}}{\|\hat{s}_j\|^2} \hat{s}_j$$

is the Gram-Schmidt orthogonalisation of $s_{k-1} = x_k - x_{k-1}$.

Proof. See [1]. □

To fix the potential (approximate) singularity of B_k we use Powell-type regularisation. This means that instead of using equation 1 to update B_k one uses the following modification

$$B_k = B_{k-1} + \frac{(\tilde{y}_{k-1} - B_{k-1}s_{k-1})\hat{s}_{k-1}^\top}{\hat{s}_{k-1}^\top s_{k-1}}.$$

Here $\tilde{y}_{k-1} = \theta_{k-1} y_{k-1} - (1-\theta_{k-1}) g_{k-1}$ where $\theta_{k-1} \in \mathbb{R}$ is chosen in dependence of a parameter $\bar{\theta}$. This modification of the update of B_k is given in algorithm 6.

If $\hat{s}_k \approx 0$ the update of B_k in $(*)$ in algorithm 6 becomes unstable and for $\hat{s}_k = 0$ ill-defined. Hence we restart the algorithm with x_k as the new starting point if

- $k = m+1$ for some fixed $m \in \mathbb{N}$ or
- $\|\hat{s}_{k-1}\| < \tau \|s_{k-1}\|$ for some fixed $\tau \in (0,1)$.

It can be shown that B_k is then well-defined.

Input : $x_0 \in \mathbb{R}^n$, $f\colon \mathbb{R}^n \to \mathbb{R}^n$, $m \in \mathbb{N}$ and $\bar{\theta}, \tau \in (0,1)$

Set $B_0 = \mathrm{Id}$, $x_1 = f(x_0)$ and $m_0 = 0$.
for $k = 1, 2, \ldots$ **do**
  Set $g_k = g(x_k)$, $m_k = m_{k-1} + 1$, $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$.
  Set $\hat{s}_{k-1} = s_{k-1} - \sum_{i=k-m_k}^{k-2} \frac{\hat{s}_i^\top s_{k-1}}{\|\hat{s}_i\|^2} s_i$.
  if $m_k = m+1$ *or* $\|\hat{s}_{k-1}\| < \tau \|s_{k-1}\|$ **then**
    Set $m_k = 0$, $\hat{s}_{k-1} = s_{k-1}$ and $B_{k-1} = \mathrm{Id}$.
  end
  Choose θ_{k-1} in dependence of $\bar{\theta}$.
  Set $\tilde{y}_{k-1} = \theta_{k-1} y_{k-1} - (1 - \theta_{k-1}) g_{k-1}$.
  Set $B_k = B_{k-1} + \frac{(\tilde{y}_{k-1} - B_{k-1} s_{k-1}) \hat{s}_{k-1}^\top}{\hat{s}_{k-1}^\top s_{k-1}}$. (*)
  Set $x_{k+1} = x_k - H_k g_k$ with $H_k = B_k^{-1}$.
end

Algorithm 6: AA-I with Powell-type regularisation and Restarting

One can then show the following Lemma.

Lemma 3 (bound on $\|H_k\|_2$)**.** *In algorithm 6 we have that H_k is well-defined and there exists a constant $c_1 > 0$ which depends on m, n, $\bar{\theta}$ and τ such that*

$$\|H_k\|_2 \leq c_1 \,.$$

Proof. See [1, Corollary 4] for details. An outline of the proof is given in Figure 1. The proof uses two lemmas which provide a bound of $\|B_k\|$ from above and a bound of $|\det B_k|$ from below. □

This bound on $\|H_k\|_2$ will later be used when showing convergence of the final algorithm. One can now replace the update formula for B_k with an update formula for H_k by the following result.

Proposition 4 (Rank-1 update for H_k)**.** *We have*

$$H_k = H_{k-1} + \frac{(s_{k-1} - H_{k-1} y_{k-1}) \hat{s}_{k-1}^\top H_{k-1}}{\hat{s}_{k-1}^\top H_{k-1} y_{k-1}}$$

This formula has been incorporated in algorithm 7. Note that the convergence behaviour of algorithm 6 is not optimal. To guarantee the decrease in $\|g_k\|$ one can interleave the AA-I steps with Krasnosel'skii-Mann (KM) steps which are given by

$$x_{k+1} = (1 - \alpha) x_k + \alpha f_k$$

for some fixed $\alpha \in (0,1)$. This yields the final algorithm 7.

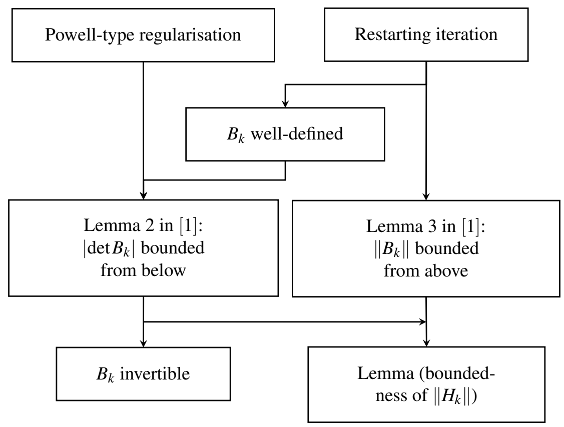

Figure 1: Outline of proof of lemma 3.

Input : $x_0 \in \mathbb{R}^n$, $f \colon \mathbb{R}^n \to \mathbb{R}^n$, $m \in \mathbb{N}$, $\bar{\theta}, \tau, \alpha \in (0,1)$ and safe-guarding constants $D, \varepsilon > 0$

Set $H_0 = \mathrm{Id}$, $x_1 = \tilde{x}_1 = f(x_0)$, $m_0 = n_{AA} = 0$ and $\bar{U} = \|g_0\|$.
for $k = 1, 2, \ldots$ **do**
  Set $g_k = g(x_k)$, $m_k = m_{k-1} + 1$, $s_{k-1} = \tilde{x}_k - x_{k-1}$ and $y_{k-1} = g(\tilde{x}_k) - g_{k-1}$.
  Set $\hat{s}_{k-1} = s_{k-1} - \sum_{i=k-m_k}^{k-2} \frac{\hat{s}_i^\top s_{k-1}}{\|\hat{s}_i\|^2} s_i$.
  if $m_k = m+1$ *or* $\|\hat{s}_{k-1}\| < \tau \|s_{k-1}\|$ **then**
    Set $m_k = 0$, $\hat{s}_{k-1} = s_{k-1}$ and $H_{k-1} = \mathrm{Id}$.
  end
  Choose θ_{k-1} in dependence of $\bar{\theta}$.
  Set $\tilde{y}_{k-1} = \theta_{k-1} y_{k-1} - (1 - \theta_{k-1}) g_{k-1}$.
  Set $H_k = H_{k-1} + \frac{(s_{k-1} - H_{k-1}\tilde{y}_{k-1})\hat{s}_{k-1}^\top H_{k-1}}{\hat{s}_{k-1}^\top H_{k-1} \tilde{y}_{k-1}}$ and $\tilde{x}_{k+1} = x_k - H_k g_k$.
  if $\|g_k\| \leq D\bar{U}(n_{AA}+1)^{-(1+\varepsilon)}$ **then**
    Set $x_{k+1} = \tilde{x}_{k+1}$ and $n_{AA} = n_{AA} + 1$.
  else
    Set $x_{k+1} = (1-\alpha)x_k + \alpha f_k$.
  end
end

Algorithm 7: AA-I with Powell-type regularisation, restarting and safeguarding

Convergence result

In this section we state and prove the main convergence result in three parts.

Theorem 5 (Convergence)**.** *Let x_k be generated by algorithm 7, then x_k converges to a fixed point of f.*

Proof. The proof follows [1, Theorem 6]. In the first part we use lemma 3 on the boundedness of $\|H_k\|_2$ and the safe-guarding step to show the convergence of g_k to 0. In part 2 we also use lemma 3 and the safe-guarding to show convergence of $a_k = \|x_k - y\|^2$ to some $a \in \mathbb{R}$. Here y denotes a fixed point of f. In the third part we use parts 1 and 2 to show that x_k converges to a fixed point.

Part 1. We partition $\mathbb{N} = K_{AA} \sqcup K_{KM}$ where $K_{AA} = \{k_0, k_1, \ldots\}$ denote the indices k where the algorithm chose an AA-step (a) and $K_{KM} = \{l_0, l_1, \ldots\}$ where the algorithm chose a KM-step (b).

if $\|g_k\| \le D\bar{U}(n_{AA}+1)^{-(1+\varepsilon)}$ **then**
  Set $x_{k+1} = \tilde{x}_{k+1}$ and $n_{AA} = n_{AA} + 1$. (a)
else
  Set $x_{k+1} = (1-\alpha)x_k + \alpha f_k$. (b)
end

Algorithm 8: The two cases for x_{k+1}.

Let y be a fixed point of f. We distinguish the cases

case (a) if $k_i \in K_{AA}$ then

$$\begin{aligned} \|x_{k_i+1} - y\| &\le \|x_{k_i} - y\| + \|H_{k_i} g_{k_i}\| \\ &\le \|x_{k_i} - y\| + c_1 \|g_k\| \\ &\le \|x_{k_i} - y\| + c_2 (i+1)^{-(1+\varepsilon)} \end{aligned} \tag{2}$$

for some constant $c_2 > 0$.

case (b) if $l_i \in K_{KM}$ then one can show that (see [1, Theorem 6])

$$\|x_{l_i+1} - y\|^2 \le \|x_{l_i} - y\|^2 - \alpha(1-\alpha)\|g_{l_i}\|^2 . \tag{3}$$

Here one uses the non-expansiveness of f and that y is a fixed point.

Hence we have in any case

$$\|x_k - y\| \le \|x_0 - y\| + c_2 \sum_i (i+1)^{-(1+\varepsilon)} = c_3 < \infty .$$

It then follows that

$$\begin{aligned} a_{k_i+1} = \|x_{k_i+1} - y\|^2 &\overset{(2),(3)}{\le} \left(\|x_{k_i} - y\| + c_2 (i+1)^{-(1+\varepsilon)} \right)^2 \\ &\le \underbrace{\|x_{k_i} - y\|^2}_{=a_{k_i}} + \underbrace{c_2^2 (i+1)^{-2(1+\varepsilon)} + 2c_2 \overbrace{\|x_{k_i} - y\|}^{\le c_3} (i+1)^{-(1+\varepsilon)}}_{=b_{k_i}} \\ &= a_{k_i} + b_{k_i} \end{aligned} \tag{4}$$

and thus

$$\alpha(1-\alpha)\sum_i \|g_{l_i}\|^2 \overset{(3)}{\leq} \sum_i a_{l_i} - a_{l_i+1} \overset{(4)}{\leq} a_0 + \sum_k b_k < \infty\,.$$

We therefore have $\lim_i \|g_{l_i}\| = 0$. It also follows from $\|g_{k_i}\| \leq D\bar{U}(i+1)^{-(1+\varepsilon)}$ that $\lim_i \|g_{k_i}\| = 0$. As all subsequences of g_k converge to 0 we thus have that g_k converges to 0.

Part 2. Let now a_{k_i} and a_{l_i} be subsequences such that $k_i \leq l_i$ and

$$a_{k_i} \xrightarrow{j\to\infty} \liminf_k a_k = \underline{a} \quad \text{and} \quad a_{l_i} \xrightarrow{j\to\infty} \limsup_k a_k = \overline{a}\,.$$

It then follows that

$$a_{l_i} - a_{k_i} = \sum_{k=k_i}^{l_i-1} a_{k+1} - a_k \overset{(4)}{\leq} \sum_{k=k_i}^{\infty} b_k$$

and when taking $l_i \to \infty$

$$\overline{a} - a_{k_i} \leq \sum_{k=k_i}^{\infty} b_k$$

and then taking $k_i \to \infty$ we get

$$\overline{a} - \underline{a} \leq 0\,.$$

Thus

$$\limsup_k a_k = \overline{a} \leq \underline{a} = \liminf_k a_k$$

and so $a_k = \|x_k - y\|^2$ converges to some $a \in \mathbb{R}$.

Part 3. Let k_i and l_i now be convergent subsequences of x_k which converge to x and $\tilde{x}$ respectively. Since by continuity of g

$$\|g(x)\| = \lim_i \|g(x_{k_i})\| \overset{\text{part 1}}{=} 0$$

we have that x is a fixed point and analogously $\tilde{x}$ is too. Now we have

$$\|x_{k_i}\|^2 = \|x_{k_i} - y\|^2 - \|y\|^2 + 2y^\top x_{k_i}$$

and by part 2 we obtain when taking $i \to \infty$

$$\|x\|^2 = a - \|y\|^2 + 2y^\top x$$

Analogously we obtain

$$\|\tilde{x}\|^2 = a - \|y\|^2 + 2y^\top \tilde{x}$$

which implies

$$2y^\top(x-\tilde{x}) = \|x\|^2 - \|\tilde{x}\|^2 .$$

As x and $\tilde{x}$ are fixed points it follows for $y \in \{x, \tilde{x}\}$ that

$$x^\top(x-\tilde{x}) = \tilde{x}^\top(x-\tilde{x})$$

and further

$$(x-\tilde{x})^\top(x-\tilde{x}) = 0 .$$

We thus have $x = \tilde{x}$. We have shown that two convergent subsequences of x_k have the same limit and hence x_k is convergent and the limit must be a fixed point of f.

□

Numerical experiments

As part of the project some numerical experiments from [1] were replicated. The obtained results are largely the same as in [1]. The aim of the experiments is to test the numerical performance of the algorithms. The functions f are chosen with Lipschitz constant close to 1 as these are precisely the type of problem for which the AA algorithm was developed.

Elastic net regression

In the first experiment f originates from an elastic net regression problem and is motivated in [1, Section 5.1f]. Specifically one obtains

$$f\colon \mathbb{R}^n \to \mathbb{R}^n, \qquad x \mapsto S_{\alpha\mu/2}\Big(x - \alpha\Big(A^\top(Ax-b) + \frac{\mu}{2}x\Big)\Big)$$

with the shrinkage operator

$$S_\kappa(x) = \big(\operatorname{sgn}(x_i)(|x_i| - \kappa)_+\big)_{i=1}^n$$

and $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ and some $\alpha, \mu \in \mathbb{R}$. Here we choose the parameters as in [1, Section 5.2], namely $m = 500$ and $n = 1000$ and A, b and x_0 to be randomly generated.

The results for the different methods can be seen in Figure 2. Here the method ‘original’ refers to the fixed point iteration, i.e. algorithm 1. The method ‘aa1-safe’ is the AA-I algorithm with Powell-type regularisation, restarting and safeguarding. The ‘aa1-matrix’ and ‘aa2-matrix’ algorithms are an implementation of the AA-I and AA-II algorithms given in 5 and 3 with limited memory. Here ‘matrix’ indicates that the implementation is not matrix-free. We see that the full-matrix implementations do not outperform the ‘original’ method and behave quite similarly. The ‘aa1’ and ‘aa1-safe’ methods behave quite similarly. Both methods however outperform the other methods on this problem.

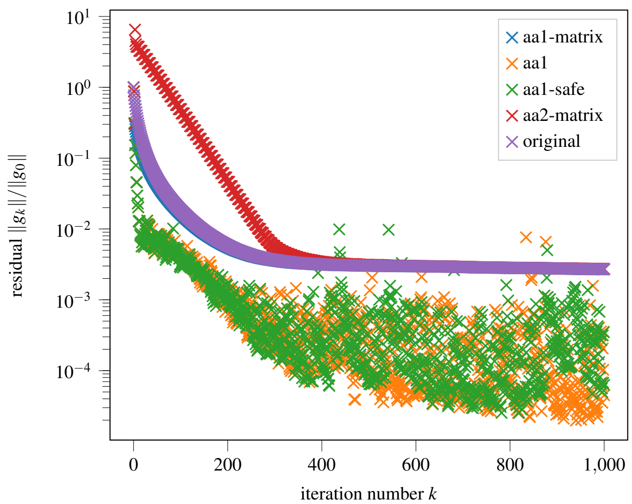

Figure 2: Residual norms for the elastic net regression problem.

Markov decision process

In a second experiment f originates from a random Markov decision process which is motivated in [1, Section 5.1f]. Our aim is to find a fixed point of the Bellman operator

$$f\colon \mathbb{R}^n \to \mathbb{R}^n, \qquad x \mapsto \left(\max_a \left(R_{sa} + \gamma \sum_{s'} P_{sas'} x_{s'}\right)\right)_{s=1}^n$$

with some $R \in \mathbb{R}^{S\times A}$, $P \in \mathbb{R}^{S\times A\times A}$ and $\gamma \in \mathbb{R}$. Here the parameter γ determines the Lipschitz-constant of f. Again, we choose the parameters as in [1, Section 5.2], namely $n = 1000$, $A = 200$, $S = 300$ and A and R to be randomly generated.

The performance for the various methods can be seen in Figure 3. In contrast to the other methods the 'aa2-matrix' method does not converge here. In this problem the fixed point iteration ('original') converges very slowly. The 'aa1-safe' method outperforms all others. This confirms numerically that the 'aa1-safe' algorithm can deal with the problems it was specifically designed for and for which the fixed point iteration fails to converge in adequate time.

In Figure 4 we see how the performance of the 'aa1-safe' method depends on the memory parameter m. In particular one sees that the algorithm performs best for this problem with a parameter of $m \approx 10$. We also see that increasing the parameter m does not necessarily improve performance of the method as in this plot the choice $m = 50$ performs worst.

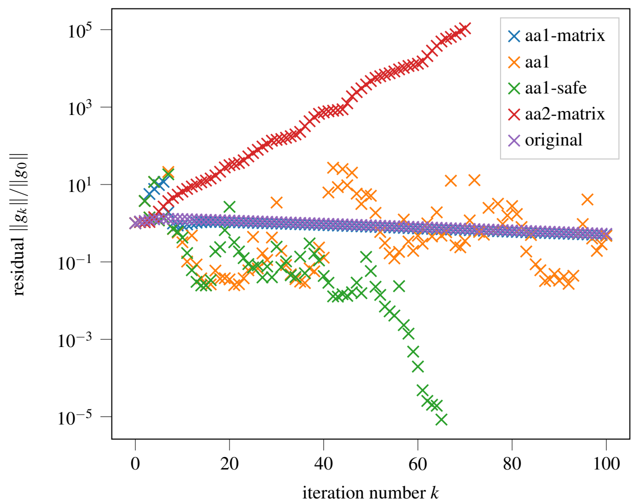

Figure 3: Residual norms for the Markov decision process problem.

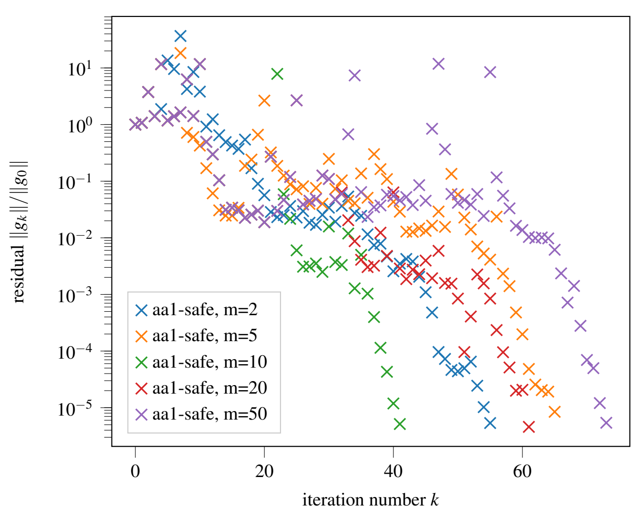

Figure 4: Residual norms for the Markov decision process problem.

Summary

The AA-I algorithm is specifically tailored to find the fixed point of a function f which is expensive to evaluate, noisy, has an unknown gradient and where the dimension n is large. The main idea of the AA-I and AA-II algorithms is to generalise the fixed point iteration by setting $x_{k+1} = \sum_i \alpha_i f_i$ for some clever choice of $\alpha = \alpha^k \in \mathbb{R}^{k+1}$. The AA-I algorithm one obtains requires some modifications. More specifically, one applies Powell-type regularisation and a restarting of the iteration for well-definedness. One builds in a mechanism for safeguarding of the steps for convergence and one uses a rank-1 update formula to make the implementation matrix-free. One can then show convergence of the algorithm under the assumption that f is non-expansive and that there exists a fixed point. The numerical experiments then show that the AA-I algorithm with the modifications outperforms the fixed point iteration for the problems tested.

Bibliography

Main source

[1] J. Zhang, B. O'Donoghue, and S. Boyd, "Globally convergent type-I Anderson acceleration for nonsmooth fixed-point iterations," *SIAM J. Optim.*, vol. 30, no. 4, pp. 3170–3197, 2020, ISSN: 1052-6234. DOI: 10.1137/18M1232772. [Online]. Available: https://doi-org.ludwig.lub.lu.se/10.1137/18M1232772.

Other sources

[2] numerics-seminar-VT23, *Github repository to the project.* Online, 2023. [Online]. Available: https://github.com/TheoKoppenhoefer/numerics-seminar-VT23.

[3] H.-r. Fang and Y. Saad, "Two classes of multisecant methods for nonlinear acceleration," *Numer. Linear Algebra Appl.*, vol. 16, no. 3, pp. 197–221, 2009, ISSN: 1070-5325. DOI: 10.1002/nla.617. [Online]. Available: https://doi-org.ludwig.lub.lu.se/10.1002/nla.617.