

Generalized Framework for Nonlinear Acceleration

Damien Scieur

Princeton University, Computer Science Department

Abstract

Nonlinear acceleration algorithms improve the performance of iterative methods, such as gradient descent, using the information contained in past iterates. However, their efficiency is still not entirely understood even in the quadratic case. In this paper, we clarify the convergence analysis by giving general properties that share several classes of nonlinear acceleration: Anderson acceleration (and variants), quasi-Newton methods (such as Broyden Type-I or Type-II, SR1, DFP, and BFGS) and Krylov methods (Conjugate Gradient, MINRES, GMRES). In particular, we propose a generic family of algorithms that contains all the previous methods and prove its optimal rate of convergence when minimizing quadratic functions. We also propose multi-secants updates for the quasi-Newton methods listed above. We provide a Matlab code implementing the algorithm.

1 Introduction

Consider the simple fixed-point iteration

$$x_{i+1} = g(x_i)$$

which produces a sequence of points $\{x_0, x_1, \dots, x_N\}$. In most cases this converges to the fixed-point x^* ,

$$x^* = g(x^*).$$

This setting is quite generic, for example in the case of optimization g can be a gradient step on an objective function f and x^* is its minimizer. The sequence $\{x_i\}$ converges to x^* at a certain speed, but ideally, we would like the procedure to be as fast as possible. For example, in optimization the accelerated gradient method [16]

$$\begin{cases} x_{i+1} &= y_i - h\nabla f(y_i) \\ y_{i+1} &= (1 + \beta)x_{i+1} - \beta x_i \end{cases}$$

converges faster to the optimum x^* than gradient method, provided good constants h and β . In practice, those constants may be hard to estimate, especially β which depends on the strong convexity parameter of the objective function, whose estimation is still a challenge [8].

Some nonlinear acceleration algorithms such as Anderson Acceleration share the same idea of Nesterov's acceleration. It combines the gradient step with a linear combination of previous iterates as follows,

$$\begin{aligned} x_i &= g(y_{i-1}), \\ y_i &= \sum_{k=1}^i \alpha_k^{(i)} x_k, \end{aligned} \tag{1}$$

where the vector $\alpha^{(i)}$ is function of the iterates, thus changing over time. As the coefficients α depend on the iterates x_i , the acceleration is thus called *nonlinear*. Its main drawbacks is the lack of convergence guarantees, and in fact it has been showed that Anderson Acceleration is unstable when $g(x)$ is not a deterministic linear function [23]. The same paper proposes a regularized version of Anderson acceleration, whose rate of convergence is asymptotically optimal even in the presence of noise [24] or when the Jacobian of g is non-symmetric [26, 2].

Other techniques such as Quasi-Newton methods schemes, popular in optimization, approximate the Newton step using the matrix $H \approx (\nabla^2 f(x_i))^{-1}$ as follow,

$$x_{i+1} = x_i - H(h\nabla f(x_i)).$$

This can be extended to fixed-point iteration by coupling a fixed-point step with a Quasi-Newton step,

$$\begin{aligned} x_i &= g(y_{i-1}), \\ y_i &= y_{i-1} - H(x_i - y_{i-1}). \end{aligned}$$

Such matrix H can be found using several formulas. The simplest ones are Broyden Type-I and Type-II updates [3], and the most popular is certainly BFGS or of DFP [17]. There also exists the symmetric rank-one update which has been rediscovered many time in many different fields.

Finally, we study Krylov subspace techniques such as the Conjugate Gradient method and GMRES [21]. These algorithms minimize some error function using a Krylov basis, usually updated with orthonormal vectors to ensure stability. Their primary usage is solving large systems of linear equations and optimizing quadratic functions.

The optimal convergence rate of Krylov methods is well-known when the fixed-point operator g is a linear mapping, and works of [26] show similar performance for Anderson Acceleration. For quasi-Newton methods, the results are less clear, even for quadratic objectives with two variables. For example, DFP and BFGS algorithms may converge poorly without line-search [20].

When the function g is nonlinear, it is unclear how fast those methods converge. In particular, the bad theoretical rates of convergence (if any) does not match the usual good numerical performance. The lack of robustness of nonlinear acceleration algorithms can explain this phenomenon since instability issues are known for some of them [19, 14, 23].

With recent result from [23, 24, 27], it is now possible to have nonlinear acceleration techniques that achieve an asymptotically optimal rate of convergence even in the presence of stochastic noise. However, because the analysis of nonlinear acceleration methods is independent of each other, we unify the analysis to identify the central argument of nonlinear acceleration.

Several results linked some acceleration methods to each other. For example [7] propose a general family of Broyden methods, including Type-I or Type-II updates, as well as Anderson mixing. Walker and Ni [28] show the link between Anderson and GMRES. However, the study does not include common schemes, such as BFGS and DFP.

Contributions. In this paper, we propose the Generalized Nonlinear Acceleration algorithm which mixes Anderson acceleration and quasi-Newton methods. In function of its parameters, it can produce the same steps than Anderson Acceleration, Broyden Type-I or Type-II, DFP, BFGS, SR- k (symmetric rank k update) or even conjugate gradients and GMRES. We give the proof of its (optimal) rate of convergence when applied to a linear function g , in the metric $\|\cdot\|_W = \|W^{1/2} \cdot\|_2$ where W is positive definite. We derive the multi-secant updates for DFP and BFGS and extends the SR-1 to SR- k updates, then analyze connections with quasi-Newton methods. We show equivalences between our algorithm and with multi-secant updates of the estimate of the Hessian. We also investigate the links with Krylov methods and propose another way to generalize CG for solving a nonlinear system of equations (or for minimizing non-quadratic functions).

Paper Organization. TODO

1.1 Notations and Assumptions

This paper studies a way to accelerate the convergence of the family of algorithms (1) when g is linear, i.e.,

$$g(x) = G(x - x^*) + x^*.$$

Usually, such mapping is written as $Ax + b$ since x^* is not explicitly known. However this notation is more convenient for our theoretical analysis. In particular, we study the following family of algorithm, that alternates between one fixed-point iteration and one linear combination step,

$$\begin{aligned} x_i &= G(y_{i-1} - x^*) + x^*, \\ y_i &= \sum_{k=1}^i \alpha_k^{(i)} x_k. \end{aligned} \tag{2}$$

Thorough this paper, we always assume that

- G is a symmetric definite positive matrix, whose spectrum is bounded by $0 \preceq G \preceq (1 - \kappa)I$ with $\kappa < 1$. Usually, κ is close to zero and often refers to be the inverse of a condition number.
- $\sum_{k=1}^i \alpha_k^{(i)} = 1$, to have a consistent algorithm [25]. It ensures $y_i = x^*$ when all x_k are replaced by x^* .
- The last coefficient $\alpha_i^{(i)}$ is nonzero. Intuitively, if this coefficient is equal to zero, we waste the last call of g , making that iteration useless.

Polynomial Notation Scieur et al. [26] shows that (2) is equivalent to a sequence of polynomials,

$$\begin{aligned} x_i &= G(y_{i-1} - x^*) + x^*, \\ y_i &= p_i(G)(y_0 - x^*) + x^*, \end{aligned} \tag{3}$$

where $p_i(G)$ is a polynomial of degree *exactly* i , whose coefficients sum to one.

1.2 Residual and Rate of Convergence

We define the residual

$$r_i = x_i - y_{i-1}. \tag{4}$$

In this paper, we often refer to the link between the residual (4) and $y_i - x^*$. In particular,

$$r_i = g(y_{i-1}) - y_{i-1} = (G - I)(y_{i-1} - x^*). \tag{5}$$

We can write this relation under the matrix form. Let the matrices, assumed to be **full column rank**,

$$\begin{aligned} X &= [x_1, x_2, \dots, x_N], \\ Y &= [y_0, y_1, \dots, y_{N-1}], \\ X^* &= x^* \mathbf{1}_N^T = [x^*, \dots, x^*] \text{ (} N \text{ times)}. \end{aligned} \tag{6}$$

In this case, the relation (5) becomes

$$R = (G - I)(Y - X^*). \tag{7}$$

We now bound the performance of algorithm (2) or equivalently (3) in the *weighted Euclidean norm*

$$\|v\|_W = v^T W v, \quad W \succ 0. \tag{8}$$

Using (3) and (5), the norm (8) of r_{i+1} can be bounded by

$$\begin{aligned} \|r_{i+1}\|_W &= \|(G - I)p_i(G)(y_0 - x^*)\|_W = \|p_i(G)r_1\|_W \\ &\leq \|p_i(G)\|_2 \|r_1\|_W. \end{aligned} \tag{9}$$

This means the performance of the algorithm (2) can be summarized by the study of $\|p_i(G)\|_2$. Ideally, we would like to find the smallest polynomial to ensure fast convergence. In this paper, we propose the Generalized Nonlinear Acceleration algorithm that finds the best polynomial in (9) at each iteration to ensure good convergence speed.

2 Generalized Nonlinear Acceleration

The Generalized Nonlinear Acceleration Algorithm 1 combines the ideas from Anderson Acceleration and quasi-Newton methods. In short, it combines linearly iterates that have been refined by a preconditioner P . This aims to minimize the residual (4) in the weighted Euclidean norm(8) defined by the weights matrix $W \succ 0$.

Data: Matrices X and Y of size $d \times N$.

Parameters: Weight matrix $W \succ 0$, Preconditioner P .

-
1. Compute matrix of residual $R = X - Y$.
 2. Solve

$$\gamma_W = \frac{(R^T W R)^{-1} \mathbf{1}_N}{\mathbf{1}_N^T (R^T W R)^{-1} \mathbf{1}_N} = \underset{\gamma: \mathbf{1}^T \gamma = 1}{\operatorname{argmin}} \|R\gamma\|_W. \quad (10)$$

3. Perform the extrapolation

$$y^{\text{extr}} = (Y - PR)\gamma_W. \quad (11)$$

Algorithm 1: Generalized Nonlinear Acceleration

In Algorithm 1, the parameters P and W are user-defined. In the next section, we discuss two standard way to choose W . Later in the paper, we show that the choice of P algorithm (1) can produce steps that are identical to existing nonlinear acceleration algorithms. For now, we can consider for simplicity that P is the scaled identity βI .

The coefficients γ_W (10), when $W = I$, correspond to Lemma 2.4 in [23]. With a minimal adaptation of the proof, this extends to arbitrary $W \succ 0$. We now quickly discuss of two "classical" choices for the weight matrix W .

2.1 Choice of W

We briefly discuss two possible choices of the weight matrix W . The first one is the simple case where $W = I$, the second one is $W = (G - I)^{-1}$.

2.1.1 Case where $W = I$

In the first case, $W = I$ simply recover the classical Anderson Acceleration [1], when $P = \beta I$ (where $\beta \neq 0$ is a scalar). This algorithm is known to minimize the residual of the extrapolation, achieving in the worst case an optimal rate of convergence [23].

2.1.2 Case where " $W = (G - I)^{-1}$ "

The case when $W = (G - I)^{-1}$ looks impossible at first sight, as it requires the inversion of $(G - I)$. However, the next proposition shows we do not need to use it explicitly.

Proposition 2.1. *Let any symmetric positive definite matrix W that satisfies*

$$WR = Y - X^*. \quad (12)$$

For example, $W = (G - I)^{-1}$. Then, the coefficients γ_W defined in Algorithm 1 can be computed using the formula

$$\gamma_W = \frac{(Y^T R)^{-1} \mathbf{1}_N}{\mathbf{1}_N^T (Y^T R)^{-1} \mathbf{1}_N}. \quad (13)$$

The proof can be found in Appendix A.1.

3 Computational Complexity of GNA

The next proposition gives the computational complexity of Algorithm 1 when matrices are updated properly. In short, for standard choices of W and P , Algorithm 1 take $O(Nd)$ operation per call. The memory size N is small in practice, thus computing y^{extr} is as costly as the step (2).

Proposition 3.1. *Assume we already have computed, using N iterates from (2),*

$$WR, \quad (R^T WR)^{-1} \mathbf{1}, \quad PR.$$

Let y^+ and r^+ be the new iterate and residual. The update

$$WR^+, \quad ((R^+)^T WR^+)^{-1} \mathbf{1}, \quad PR^+.$$

costs at most $O_W + O_P + O(Nd + N^2)$ operations, where O_W and O_P are upper bounds on the number of operations for WR^+ and PR^+ .

Proof.

First, we update the matrix $R^T WR$ to $[R, r^+]^T W [R, r^+]$. After expansion,

$$[R, r^+]^T W [R, r^+] = \begin{bmatrix} R^T WR & R^T W r^+ \\ (r^+)^T WR & (r^+)^T W r^+ \end{bmatrix}$$

Since WR and $R^T WR$ are known, the complexity of the update is equal to $O(O_W)$ (computation of the bottom-right element) plus $O(Nd)$.

The second step produces the coefficients γ_W by solving the system of equations $([R, r^+]^T W [R, r^+])^{-1} \mathbf{1}_N$. The size of the system grows in N , which means $O(N^3)$ operation to solve it. However, we update the matrix $R^T WR$ with a rank two matrix. With the Woodbury matrix identity (Appendix B.1) it is possible to update the previous solution γ_W , reducing the complexity to $O(N^2)$.

Finally, the last step consists in forming the extrapolation. First, it takes $O(dN)$ operation to form $Y\gamma_W$ and $R\gamma_W$. Then, we apply the preconditioner P to $R\gamma_W$, which takes $O(O_P)$ operations.

The total complexity is thus bounded by

$$O_W + O_P + O(Nd + N^2).$$

■

In practice, N is much smaller than d , and usually $N \in [5, 20]$ in most application. In addition, the matrix W is usually used implicitly (see section 2.1), so $O_W = O(1)$. Finally, the preconditioner P corresponds to the sum of the identity matrix and a rank N matrix in most nonlinear acceleration algorithms, thus $O_P = O(N^2 d)$. Under those conditions, the computational complexity of GNA is bounded by $O(N^2 d)$ where N^2 is small compared to d . This property is desirable as it is *as expansive as* computing a new residual r^+ .

4 Rate of Convergence

4.1 Chebyshev Acceleration

We have seen that the norm of the polynomial (9) quantifies the rate of convergence of algorithm (3),

$$\|r_{i+1}\|_W \leq \|p_i(G)\|_2 \|r_1\|_W,$$

where p_i is a polynomial of degree exactly equal to i whose coefficients sums to one. The best polynomial of this class is a rescaled Chebyshev polynomial [11], which achieves the optimal rate

$$\|r_{i+1}\|_W \leq \frac{\xi^i}{1+\xi^{2i}} \|r_1\|_W, \quad \xi = \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}} \quad (14)$$

where $\|G\|_2 \leq 1 - \kappa$. This requires the knowledge of κ , usually referring to the inverse of a condition number and thus unknown in practice. In addition, Chebyshev acceleration is not adaptive to the initial point r_1 . The next section shows that Algorithm 1 achieves the same bound without the knowledge of κ .

4.2 Optimal Rate of Convergence of (offline) GNA

The next Theorem shows Algorithm 1 implicitly solves

$$\min_p \|p(G)r_1\|_W \quad \text{s.t. } p(1) = 1, \quad \deg(p) \leq N - 1.$$

This strategy is optimal, in the sense that it recovers (14) up to a constant that depends on the preconditionner P .

Theorem 4.1. *Let $\{(x_1, y_0), \dots, (x_N, y_{N-1})\}$ be pairs generated by (2) (or equivalently (3)) where the matrix G is symmetric and $\|G\| \leq 1 - \kappa$. Let y^{extr} be the output of Algorithm 1. Consider the residual of the extrapolation*

$$r^{extr} = g(y^{extr}) - y^{extr}.$$

Then, the norm $\|r^{extr}\|_W$ is bounded by

$$\|r^{extr}\|_W \leq \|I - (G - I)P\|_2 \min_{\substack{\deg(p) \leq N-1 \\ p(1)=1}} \|p(G)r_1\|_W \quad (15)$$

$$\leq \|I - (G - I)P\|_2 \frac{\xi^{N-1}}{1 + \xi^{2(N-1)}} \|r_1\|_W \quad (16)$$

where ξ is defined in (14) and W is a positive definite. In addition, after at most d steps, $r^{extr} = 0$.

The proof can be found in Appendix A.2. This theorem shows that it is possible to ensure the optimal rate of convergence by post-processing the iterates using Algorithm (1) for any norm defined by W . Also, a finite number of iteration are required to reach x^* .

Intuitively, it would be more efficient to inject at each iteration the output of GNA. The next section studies this strategy and gives a sufficient condition to ensure an optimal rate of convergence.

4.3 Online GNA

As discussed in the previous section, instead of computing y^{extr} on the side, we can inject the extrapolated point directly in (2) as following,

$$\begin{aligned} x_i &= G(y_{i-1} - x^*) + x^*, \\ y_i &= y^{extr}. \end{aligned} \quad (17)$$

However, it is not clear from Algorithm (1) that y^{extr} satisfies the special structure (3), in particular we need to ensure that the extrapolation can be written as a polynomial in the matrix G , whose degree is exactly equal to $i - 1$ and its coefficients sums to one. The next proposition gives a simple condition on P to ensure those two properties.

Proposition 4.2. *Consider the output of Algorithm 1 after N iterations of (3). If we have, for $\beta \neq 0$,*

$$y^{extr} = (Y - PR)\gamma_W = Y\tilde{\gamma} + \beta R\gamma_{\tilde{W}},$$

where $\tilde{\gamma}$ is an arbitrary vector whose coefficients sum to one, and \tilde{W} a positive definite matrix, then $y^{extr} - x^* = p_{N-1}(G)(y_0 - x^*)$, where p_N is a polynomial of degree $N - 1$ whose coefficients sum to one.

The proof can be found in Appendix A.3. In short, this proposition shows that whatever the coefficients that combine the y_i 's, as long as we use some "optimal" coefficients for combining the residuals, the extrapolation satisfies the structure (3). By using this argument recursively, (17) produces iterates with the same structure than (3). In combination with Theorem 4.1, this proves the optimality of the rate of convergence of online Generalized Nonlinear Acceleration algorithm.

It remains to fix the value of P . The next sections show some choices of the preconditioner that correspond to existing nonlinear acceleration algorithms.

5 Connections with Anderson Acceleration

We begin with nonlinear acceleration algorithms that form a polynomial that minimizes (9) at each iteration. The link between GNA and this class of algorithm is quite straightforward as they share the same ideas.

The Anderson Acceleration [1], Minimal Polynomial Method [4] and Mesina Method [15] are different variant of the same algorithm. They solve, at each iteration,

$$\gamma_{\text{Anderson}} = \underset{\gamma: \mathbf{1}^T \gamma = 1}{\operatorname{argmin}} \|R\gamma\|_2.$$

Anderson Acceleration and Mesina method use the same formula, while MPE method uses a slightly different approach giving the same result. Then, Anderson acceleration uses γ_{Anderson} to combines the previous iterates using a so-called *mixing* parameter $\beta \neq 0$ as following,

$$y^{\text{extr}} = (Y - \beta R)\gamma_{\text{Anderson}}.$$

Clearly, this corresponds to a special case of Algorithm 1 where $W = I$ and $P = \beta I$.

Recently, Fang and Saad [7] introduce the type-I (or "Good") Anderson Acceleration. They used the Broyden Type-I update to derive the formula

$$\gamma_{\text{Good Anderson}} = \frac{(Y^T R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T R)^{-1} \mathbf{1}}. \quad (18)$$

Then, it combines the previous iterates as follow,

$$y^{\text{extr}} = (Y - \beta R)\gamma_{\text{Good Anderson}}.$$

This formula comes from the analogy between Anderson and Broyden methods, so it is unclear why (18) is a good candidate for minimizing the norm of the residual (9). Besides, the rate of convergence was not specified.

With our results, we see that equation (18) corresponds to (10) when $W = (G - I)^{-1}$. This means equation (18) is the solution of (10). With the application of Theorem 4.1, we directly obtain the rate of convergence of the Good Anderson Acceleration. Indeed, using GNA with $W = (G - I)^{-1}$ and $P = \beta I$ produces the same extrapolated point, thus leading to an optimal rate of convergence.

6 Connections with Quasi-Newton (qN)

We quickly introduce the idea of quasi-Newton methods. First, consider Newton methods step

$$y_{\text{Newton}} = y_{N-1} - (G - I)^{-1} r_N.$$

Using equation (5), we have $y_{\text{Newton}} = x^*$. However it needs $O(d^3)$ operations to invert the matrix $(G - I)$.

Unlike the Newton method, the qN step is cheaper but does not converge in one iteration. It uses an approximation $H \approx (G - I)^{-1}$ and performs

$$y_{\text{q-Newton}} = y_{N-1} - H r_N, \quad (19)$$

where the matrix H satisfies *secant equations*

$$H(r_{i+1} - r_i) = (y_i - y_{i-1}) \quad \forall i = 1 \dots N - 1.$$

In the matrix form, this gives

$$HRC = YC, \quad \mathbf{1}^T C = 0, \quad \mathbf{rank}(C) = N - 1. \quad (20)$$

For example, C can be formed using $N - 1$ different vectors $[\dots, 0, 1, -1, 0, \dots]^T$. Indeed, the secant equation still holds if H is replaced by $(G - I)$. Some qN methods estimates instead the matrix $J \approx G - I$ then take its inverse $H = J^{-1}$. In this case, J follows

$$RC = JYC, \quad \mathbf{1}^T C = 0, \quad \mathbf{rank}(C) = N - 1. \quad (21)$$

Any H (or J^{-1}) that satisfies the secant equation (20) is a candidate for the qN step. There exist several strategies to choose one particular matrix. In the next section, we introduce the generalized quasi-Newton step, then we investigate further more classical schemes.

6.1 Generalized qN Method

We introduce here the Generalized qN scheme, which performs the extrapolation

$$y_{\text{Generalized qN}} = (Y - HR)\gamma, \quad \forall \gamma^T \mathbf{1} = 1, \quad (22)$$

where γ is an arbitrary vector whose entries sum to one and H satisfies the secant equation (20). Proposition A.3 shows that (22) is invariant in the choice of γ . For illustration, classical schemes use $\gamma = [0, \dots, 0, 1]$.

The solution of the secant equation (20) is given by

$$H = YC(RC)^\dagger + H_0 RC(RC)^\dagger, \quad (23)$$

where $(RC)^\dagger$ is the generalized inverse of RC and H_0 is arbitrary (it can be viewed as the initialization). There exist a large number of generalized pseudo-inverses, and choosing one of them corresponds to one qN method.

Some qN schemes minimize the distance between H and H_0 in, for example, the *weighted Frobenius norm*

$$\|A\|_M = \text{Trace}(M^{1/2} A^T M A M^{1/2}), \quad M \succ 0. \quad (24)$$

Other methods impose in addition some constraints on the matrix H , for example symmetry.

Despite the different approach of qN and GNA, writing the qN method under the form of (22) makes the links straightforward: (22) is equivalent to run Algorithm 1 with $P = H$ and W arbitrary. Using Theorem 4.1, this means that using *any* matrix that satisfies the secant equation (20) in the qN step (22) lead to an optimal rate of convergence.

However, it is unclear if a given qN method meets the requirements of Proposition 4.2. In addition, the computation of (23) may be complicated in some cases. Finally, it is unclear how bad is the constant factor in Theorem 4.1.

In the next section, we show that specific choices of W (and thus γ_W) simplify the term $HR\gamma$ in (22) for some known qN schemes. It leads to more explicit update rules that meet the assumptions of Proposition 4.2.

6.2 Broyden Methods

6.2.1 (Multi-Secant) Type-I Broyden

The Type-I Broyden method estimates the matrix $G - I$ first, then invert it. Among all the possible solutions of (21) it chooses the closest to the initialization J_0 in the Frobenius norm. Here, we consider the general case in

the weighted Frobenius norm, i.e.,

$$J = \operatorname{argmin}_J \|J - J_0\|_M, \quad JYC = RC,$$

then $H = J^{-1}$ using the Woodbury identity (See Appendix B.1). The explicit formula for H is derived in Appendix C.2. After simplification, injecting H in (22) gives

$$y_{\text{Type-I Broyden}} = (Y - J_0^{-1}R)\gamma_{\tilde{M}} \quad (25)$$

$$\text{Where } \tilde{M} = (G - I)^{-1}M^{-1}J_0^{-1}. \quad (26)$$

This corresponds to Algorithm 1 with parameters $P = J_0^{-1}$ and $W = \tilde{M}$. At this state, this is unclear if \tilde{M} is symmetric definite definite, so Theorem 4.1 does not apply in this situation. If we consider the simple case where $M = I$ and $J_0^{-1} = \beta I$ with $\beta \neq 0$, (25) simplifies into

$$y_{\text{Type-I Broyden}} = (Y - \beta R)\gamma_{(G-I)^{-1}}. \quad (27)$$

This corresponds to Algorithm 1 with $P = \beta I$ and $W = \beta(G - I)^{-1}$. With these parameters Theorem (4.1) shows that the Broyden Type-I method is an optimal algorithm for minimizing the residual in the norm $\|\cdot\|_{(G-I)^{-1}}$. In addition, the step meets the structure of Proposition 17, so online acceleration is possible.

6.2.2 (Multi-Secant) Type-II Broyden

The Type-II Broyden method estimates directly $(G - I)^{-1}$ by choosing H as close as possible to the initialization H_0 in the Frobenius norm. Again, we give here the update in the weighted Frobenius norm (24) (see Appendix C.3 for more details). We have

$$H = \operatorname{argmin}_H \|H - H_0\|_{M^{-1}} \quad \text{s.t. } HRC = YC.$$

After simplification, injecting H in (22) gives

$$y_{\text{Type-II Broyden}} = (Y - H_0R)\gamma_M.$$

This time, since M is symmetric positive definite, Theorem 4.1 directly applies, but this is not the case with Proposition (17). However if we assume $H_0 = \beta I$ with $\beta \neq 0$, online acceleration is possible since

$$y_{\text{Type-II Broyden}} = (Y - \beta R)\gamma_I. \quad (28)$$

6.2.3 Observations

Usually, the matrices J_0 and H_0 correspond to the previous estimate and are updated with rank-one matrices. However, for the Type-I method, it is unclear if the method has an optimal rate of convergence, as (26) may not be symmetric positive definite. Thus Theorem 4.1 does not apply.

When the initialization is a scaled identity matrix, The Broyden Type-I (27) and Broyden Type-II (28) steps are equivalent up to the weight matrix M . For example, the Broyden Type-2 updates with $M = (G - I)^{-1}$ is identical to Broyden Type-I with $M = I$. In this case, they are also equivalent to Anderson Acceleration.

6.3 Symmetric Methods

We introduce the multi-secant symmetric updates. This setting was studied by [22] for DFP and BFGS (not SR- k), but not for the general weighted Frobenius norm. It requires that (21) admits a symmetric solution, or equivalently that [6, 13]

$$(YC)^T RC = (RC)^T (YC).$$

This may not be the true when g in (1) is nonlinear, but in our setting (2) the symmetric solution exists since

$$(YC)^T RC = (YC)^T (G - I)YC = (RC)^T (YC).$$

6.3.1 (Multi-Secant) DFP

The DFP Algorithm [5, 9] finds the closest *symmetric* matrix to J_0 in the weighted Frobenius norm defined by the matrix $(G - I)^{-1}$ that solves (21). In Appendix C.4 we consider the general case

$$J = \operatorname{argmin}_J \|J - J_0\|_M, \quad JYC = RC, \quad J = J^T, \quad (29)$$

where M is arbitrary. The explicit formula (see Appendix C.4) is more complicated than Broyden updates, so we directly jump to the standard and simpler case where

$$M = (G - I)^{-1}, \quad J_0^{-1} = \beta I.$$

Injecting the solution of (29) in the qN step (22) gives

$$y_{\text{DFP}} = Y\gamma_I - \left(\beta I + YC((YC)^T RC)^{-1}(YC)^T \right) R\gamma_I$$

This is equivalent to Algorithm 1 with parameters $W = I$ and $P = \beta I + YC((YC)^T RC)^{-1}(YC)^T$. This step can be written in the form of Propoposition (4.2),

$$Y \underbrace{\left((I - C((YC)^T RC)^{-1}(YC)^T R) \right)}_{=\tilde{\gamma}} \gamma_I + \beta R\gamma_I.$$

By Proposition 4.2 and Theorem 4.1, online acceleration is possible if $\beta \neq 0$ with an optimal convergence rate.

6.3.2 (Multi-Secant) BFGS

We now introduce the BFGS algorithm. The idea is similar to the DFP formula, but it updates the approximation $(G - I)^{-1}$ directly. Again, solving the general case

$$\operatorname{argmin}_H \|H - H_0\|_{M^{-1}} \quad \text{s.t.} \quad HRC = YC, \quad H = H^T$$

leads to complicated formula (see Appendix C.5 for details). We directly jump to the standard case where

$$M = (G - I)^{-1}, \quad H_0 = \beta I.$$

Injecting H in the generalized qN step (22) gives

$$\begin{aligned} y_{\text{BFGS}} &= Y\gamma_{(G-I)^{-1}} - \beta R\gamma_{(G-I)^{-1}} \\ &+ \beta YC((YC)^T RC)^{-1}(RC)^T R\gamma_{(G-I)^{-1}}. \end{aligned}$$

This corresponds to Algorithm 1 with parameters

$$W = (G - I)^{-1}, \quad P = \beta(I - YC((YC)^T RC)^{-1}(RC)^T).$$

With the initialization $H_0 = \beta I$ we can use online acceleration (Proposition 4.2). Unlike DFP, the usage of β here is similar to (27) and (28).

6.3.3 SR-1 and SR- k

We now introduce the symmetric rank k (SR- k) update, whose SR-1 is a particular member. The method starts with an initialization H_0 , then looks to a symmetric, low-rank update $H - H_0$ such that H satisfies the secant equation (20). More formally, SR- k solves

$$H = \min \operatorname{rank}(H - H_0) \quad \text{s.t.} \quad HRC = YC, \quad H = H^T.$$

Its explicit solution is given in Appendix C.6, and injecting this matrix in the generalized qN step (22) gives

$$y_{\text{SR-}k} = (Y - H_0 R) \gamma_{(G-I)^{-1} - H_0}. \quad (30)$$

The vector $\gamma_{(G-I)^{-1} - H_0}$ is computed using the formula

$$\gamma_{(G-I)^{-1} - H_0} = \frac{(Y^T R - R^T H_0 R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T R - R^T H_0 R)^{-1} \mathbf{1}}.$$

We have here a direct correspondence between the SR- k update and Algorithm 1 with

$$P = H_0, \quad W = (G - I)^{-1} - H_0.$$

Under the condition that $(G - I)^{-1} \succ H_0$, Theorem (4.1) applies, so the SR- k updates has an optimal rate of convergence. If $H_0 = \beta I$, by proposition 4.2 the SR- k algorithm (30) can be used online.

6.4 Comparison with Previous Work

We showed, with Theorem 4.1, that qN methods with multi-secant updates converge with a rate similar to Chebyshev acceleration (14). In addition, after at most d iterations they reach exactly x^* . In practice, quasi-Newton methods are used with a single secant update, so our convergence result may not apply, as (20) may not hold for all secants.

There exist some convergence results for single secant update of some quasi-Newton methods. For example, Gay [10] shows that Broyden Type-I and Type-II update converges to x^* after $2d$ steps. Nocedal and Wright [17] show that BFGS and DFP with *with exact line-search* converges after d steps, but it does not hold for inexact line search or unitary step size. Moreover, the rate of convergence for the first N iterations is still unknown, and in fact can be particularly bad even for a quadratic problem with two variables [20].

Surprisingly, the SR-1 method converges after d iterations [17] because it satisfies all previous secant equations. Here, with Theorem 4.1 we refine the result by specifying its rate on quadratics for any $N < d$.

A priori, there is no reason to use single-secant updates for optimizing quadratics as they are as costly as the multi-secant update, and their theoretical properties are apparently weaker. This may not be true numerically, so we compare the two approaches in Appendix E.

7 Connections with Krylov Methods

We investigate connections with Krylov methods, used for solving linear systems $Ax = b$ where A is positive definite, or equivalently minimizing a quadratic function. In the case where $g(x)$ is a linear function (2), we can also use Krylov methods to find an extrapolation y^{extr} .

Krylov methods start with an approximation x_0 , associated to the residual r_1 , then generate the *Krylov subspace*

$$\begin{aligned} \mathcal{K}_N &= \text{span}\{\underbrace{[r_1, Gr_1, G^2 r_1, \dots, G^{N-1} r_1]}_{=K_N}\} \\ &= \{q(G)r_1 : \deg(q) \leq N - 1\}. \end{aligned} \quad (31)$$

As the *Krylov matrix* K_N can be ill-conditioned in practice, Krylov subspace methods maintain an orthonormal basis to \mathcal{K}_N to ensure better numerical stability.

After creating the subspace, Krylov methods minimize some error function $e(x)$, like the norm of the residual, under the constraint that the next iterate belongs to $x_0 + \mathcal{K}_N$,

$$y^{\text{extr}} = \min_{x \in x_0 + \mathcal{K}_N} e(x). \quad (32)$$

Different error functions lead to different Krylov methods.

7.1 MINRES and GMRES

In the case of GMRES (or equivalently MINRES, since we work with symmetric matrices), the iterations are *not* coupled. The algorithm creates a basis K_N of \mathcal{K}_N using Arnoldi (MINRES, [18]) or Lancos (GMRES, [21]) then computes

$$y_{\text{GMRES}} = \operatorname{argmin}_{x \in x_0 + \mathcal{K}_{N-1}} \|x - g(x)\|_2.$$

We see the iterates belongs to \mathcal{K}_{N-1} rather than \mathcal{K}_N . This can be explained by the fact that the residual of y_{GMRES} belongs to \mathcal{K}_N . This corresponds to (32) with $e(x) = \|\cdot\|_2$. In Appendix D.1, we show that

$$y_{\text{GMRES}} = x^* + p^*(G)(x_0 - x^*), \quad p^* = \operatorname{argmin}_{\substack{\deg(p) \leq N-1, \\ p(1)=1.}} \|p(G)r_1\|_2.$$

Even if y_{GMRES} belongs to $x_0 + \mathcal{K}_{N-1}$, the algorithm uses \mathcal{K}_N to compute the polynomial p^* . We can cast GMRES into an instance of GNA using $P = 0$, $W = I$. Indeed,

$$y_{\text{GNA}} = Y\gamma_I = x^* + (Y - X^*)\gamma_I = x^* + (G - I)^{-1}R\gamma_I$$

Appendix D.1 shows that $R\gamma_I = p^*(G)r_1$, thus

$$y_{\text{GNA}} = y_{\text{GMRES}} \quad \text{when } W = I, \quad P = 0.$$

Thus, GNA and GMRES / MINRES are equivalent. Since $P = 0$, by Proposition 4.2, these algorithms are not suitable for online acceleration.

7.2 Conjugate Gradient

The Conjugate Gradient in an online acceleration algorithm, with is usually represented under the form of a two-step recurrence. Conceptually, CG builds a new iterate whose residual is orthogonal to the search space (31). This can be written under the form of (32),

$$y_{\text{CG}} = \operatorname{argmin}_{y \in x_0 + \mathcal{K}_N} \|(K_N)^T(G - I)(y - x^*)\|_2$$

Appendix D.2 shows that y_{CG} is obtained using GNA with

$$P = R(R^T(G - I)R)^{-1}R^T \quad \text{and} \quad W \text{ arbitrary.} \quad (33)$$

Here we see that we need to multiply R with $G - I$, which implies two call to $g(x)$ per iteration. Appendix D.2 shows how to avoid this problem using the linearity of g by deducing the residual of y_{CG} . With this strategy, the matrix $R^T R$ is diagonal and $R^T(G - I)R$ tri-diagonal. The solution can thus be updated using a two-step recurrence.

Conjugate Broyden Method Using (33) with $W = I$, the iterate of GNA simplifies into

$$y_{\text{GNA}} = Y\gamma_I + \beta^* R\gamma_{G-I}, \quad \beta^* = \frac{\mathbf{1}^T (R^T(G-I)R)^{-1} \mathbf{1}}{\mathbf{1}^T (R^T R)^{-1} \mathbf{1}}. \quad (34)$$

In Appendix D.2, we show that β^* can be found by solving

$$\beta^* = \operatorname{argmin}_{\beta, z} f(z), \quad z = Y\gamma_I + \beta R\gamma_{G-I},$$

where $f(z)$ is the quadratic objective function whose gradient step is represented by g . This algorithm a possible extension of nonlinear conjugate gradient. The computation of γ_{G-I} requires one Hessian-vector operation, or an approximation of $G - I$. Then, one line-search on the objective function f is needed to compute β .

Compared to other nonlinear CG algorithm, it requires more memory ($O(N)$ rather than the two previous iterations), but this looks more stable as it does not relies on too much on the linearity of g . We present some numerical experiments on nonlinear functions in Appendix E.

8 Numerical Experiments

We briefly compare several instances of GNA for optimizing a quadratic function. We minimize the linear regression

$$\min_x \frac{1}{2} (\|Ax - b\|_2^2 + \lambda \|x\|_2^2)$$

using the fixed-step gradient method for strongly convex functions [16], combined with GNA using online acceleration (17).

We use data sets **Rand** and **Madelon** on all several instances of GNA algorithm. For dataset **Rand**, A is generated randomly using Gaussian distribution and the vector y is full of ones. Its size is 50×25 . We use $N = \infty$, i.e., N grows linearly with i . This experiment highlights the results of Theorem 4.1, i.e., optimal rate and convergence after d iterations of GNA.

The second dataset, **Madelon** comes from Guyon [12], and posses 2000 data point in dimension 500. Here, we fix $N_{\max} = 20$. This experiment illustrates the efficiency of previous methods when used with limited memory.

We distinguish the case with and without line-search on the β , the mixing parameter in Anderson acceleration, or the initialization $H_0 = \beta I$ in qN methods. Without line-search, the parameter is fixed to -1 . The starting point x_0 is full of ones and λ is set such that $\kappa = 10^{-6}$. The experiments are presented in Figure 1. Since GMRES and CG are a bit different (the line search is necessary, and they do not (2)), we do not compare them in this section. We performed more advanced numerical experiments in Appendix E.

We see in Figure 1 that, when used with full memory, all algorithms behave the same. This is still true when GNA is used with limited memory and without line-search, with a slight advantage to BFGS and Bad Broyden/Anderson method. With line-search, BFGS and Bad Broyden/Anderson methods are superior to the others, which explains why BFGS is more used in practice.

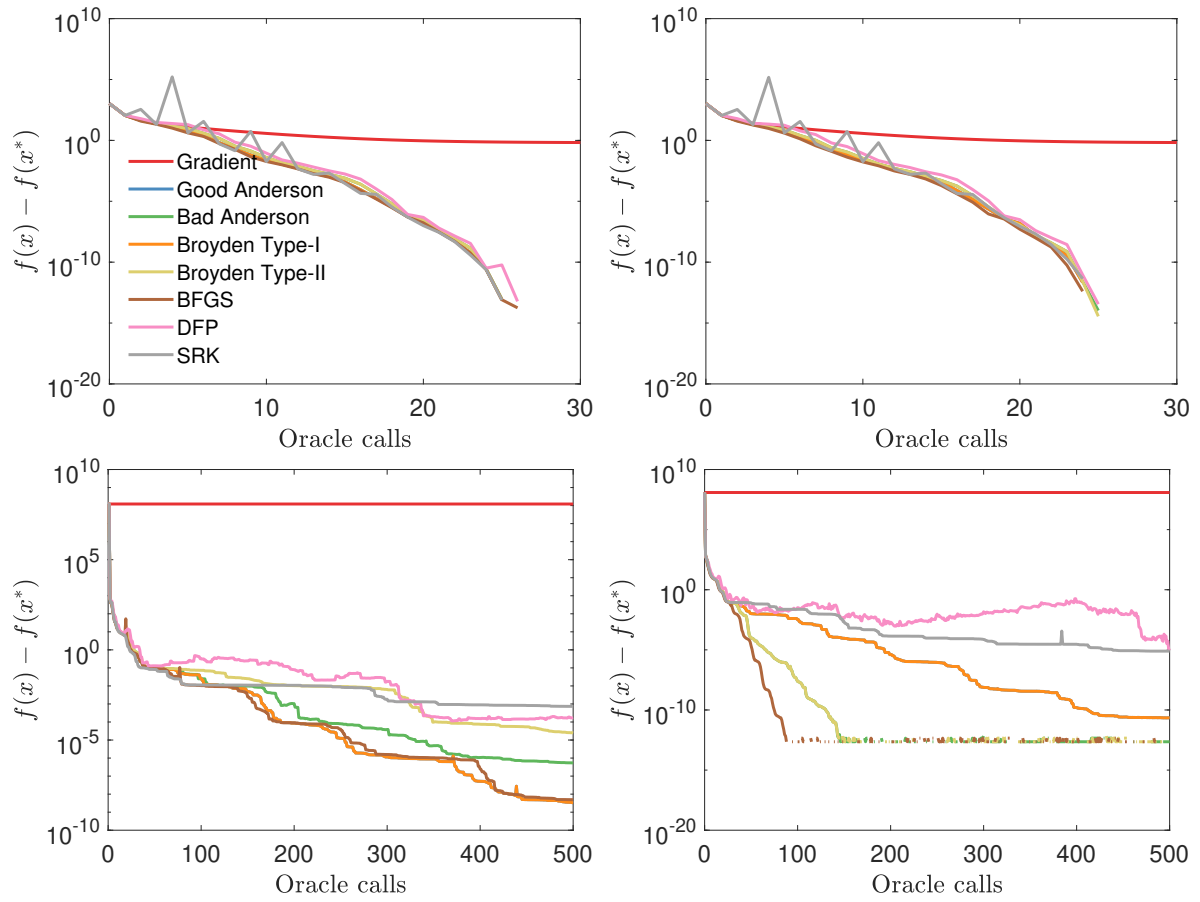


Figure 1: Comparison between different nonlinear acceleration algorithm. Top row: Rand, bottom: Sido0. Left column is with fixed step size, right with line-search.

References

- [1] Donald G Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, 1965.
- [2] Raghu Bollapragada, Damien Scieur, and Alexandre d’Aspremont. Nonlinear acceleration of momentum and primal-dual algorithms. *arXiv preprint arXiv:1810.04539*, 2018.
- [3] Charles G Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of computation*, 19(92):577–593, 1965.
- [4] Stan Cabay and LW Jackson. A polynomial extrapolation method for finding limits and antilimits of vector sequences. *SIAM Journal on Numerical Analysis*, 13(5):734–752, 1976.
- [5] William C Davidon. Variable metric method for minimization. *SIAM Journal on Optimization*, 1(1):1–17, 1991.
- [6] FJ Henk Don. On the symmetric solutions of a linear matrix equation. *Linear Algebra and its Applications*, 93:1–7, 1987.
- [7] Haw-Ren Fang and Yousef Saad. Two classes of multiseant methods for nonlinear acceleration. *Numerical Linear Algebra with Applications*, 16(3):197–221, 2009.
- [8] Olivier Fercoq and Zheng Qu. Restarting accelerated gradient methods with a rough strong convexity estimate. *arXiv preprint arXiv:1609.07358*, 2016.
- [9] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.
- [10] David M Gay. Some convergence properties of broyden’s method. *SIAM Journal on Numerical Analysis*, 16(4):623–630, 1979.
- [11] Gene H Golub and Richard S Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods. *Numerische Mathematik*, 3(1):157–168, 1961.
- [12] Isabelle Guyon. Design of experiments of the nips 2003 variable selection benchmark, 2003.
- [13] DAI Hua. On the symmetric solutions of linear matrix equations. *Linear Algebra and its Applications*, 131:1–7, 1990.
- [14] Duane D Johnson. Modified broyden’s method for accelerating convergence in self-consistent calculations. *Physical Review B*, 38(18):12807, 1988.
- [15] M Mešina. Convergence acceleration for the iterative solution of the equations $x = ax + f$. *Computer Methods in Applied Mechanics and Engineering*, 10(2):165–173, 1977.
- [16] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.
- [17] Jorge Nocedal and Stephen J Wright. *Nonlinear Equations*. Springer, 2006.
- [18] Christopher C Paige and Michael A Saunders. Solution of sparse indefinite systems of linear equations. *SIAM journal on numerical analysis*, 12(4):617–629, 1975.
- [19] Michael James David Powell. Restart procedures for the conjugate gradient method. *Mathematical programming*, 12(1):241–254, 1977.
- [20] MJD Powell. How bad are the bfgs and dfp methods when the objective function is quadratic? *Mathematical Programming*, 34(1):34–47, 1986.

- [21] Youcef Saad and Martin H Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869, 1986.
- [22] Robert B Schnabel. Quasi-newton methods using multiple secant equations. Technical report, Colorado University, 1983.
- [23] Damien Scieur, Alexandre d’Aspremont, and Francis Bach. Regularized nonlinear acceleration. In *Advances In Neural Information Processing Systems*, pages 712–720, 2016.
- [24] Damien Scieur, Francis Bach, and Alexandre d’Aspremont. Nonlinear acceleration of stochastic algorithms. In *Advances in Neural Information Processing Systems*, pages 3985–3994, 2017.
- [25] Damien Scieur, Vincent Roulet, Francis Bach, and Alexandre d’Aspremont. Integration methods and optimization algorithms. In *Advances in Neural Information Processing Systems*, pages 1109–1118, 2017.
- [26] Damien Scieur, Edouard Oyallon, Alexandre d’Aspremont, and Francis Bach. Nonlinear acceleration of deep neural networks. *arXiv preprint arXiv:1805.09639*, 2018.
- [27] Damien Scieur, Edouard Oyallon, Alexandre d’Aspremont, and Francis Bach. Nonlinear acceleration of CNNs. In *Workshop of International Conference on Learning Representations*, 2018.
- [28] Homer F Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.
- [29] Max A Woodbury. Inverting modified matrices. *Memorandum report*, 42(106):336, 1950.

A Missing Proofs

A.1 Proof of Proposition 2.1

Proposition A.1. *Let the symmetric positive definite matrix W satisfies (12). Then, the coefficients γ_W defined in Algorithm 1 can be computed using the formula*

$$\gamma_W = \frac{(Y^T R)^{-1} \mathbf{1}_N}{\mathbf{1}_N^T (Y^T R)^{-1} \mathbf{1}_N}. \quad (35)$$

Proof. We start with (10). If we expand the objective function we have

$$\gamma^T (Y - X^*) (G - I)^{-1} (Y - X^*) \gamma.$$

However, $\gamma^T X^* = x^*$ since $\gamma^T \mathbf{1} = 1$. If we drop the constant terms, the objective function becomes

$$\gamma^T Y^T (G - I) Y \gamma + 2\gamma^T Y (G - I) x^*.$$

Now, consider the Lagrangian function $\mathcal{L}(\gamma, \pi)$,

$$\gamma^T Y^T (G - I) Y \gamma + 2\gamma^T Y (G - I) x^* + 2\pi(\gamma^T \mathbf{1} - 1).$$

Its derivative over γ gives

$$\frac{\nabla_{\gamma} \mathcal{L}(\gamma, \pi)}{2} = Y^T (G - I) Y \gamma + Y (G - I) x^* + \pi \mathbf{1}$$

Using $X^* \gamma = x^*$, we can simplify it into

$$\frac{\nabla_{\gamma} \mathcal{L}(\gamma, \pi)}{2} = Y^T (G - I) (Y - X^*) \gamma + \pi \mathbf{1}.$$

Finally, since $(G - I)^{-1} (Y - X^*) = R$ we have

$$\frac{\nabla_{\gamma} \mathcal{L}(\gamma, \pi)}{2} = Y^T R \gamma + \pi \mathbf{1}.$$

If we equal the derivative to zero,

$$\gamma = -\pi (Y^T R)^{-1} \mathbf{1}.$$

Finally, using the constraint that $\gamma^T \mathbf{1} = 1$, we have

$$\pi = \frac{-1}{\mathbf{1}^T (Y^T R)^{-1} \mathbf{1}}.$$

■

This proposition can be easily extended to the general case there $W = (G - I)^{-1} W_1 + W_2$, assuming $(G - I)^{-1} W_1 + W_2$ symmetric positive definite. In this case,

$$\gamma_W = \frac{(Y^T W_1 R + R^T W_2 R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T W_1 R + R^T W_2 R)^{-1} \mathbf{1}}. \quad (36)$$

A.2 Proof of Theorem 4.1

Theorem A.2. Let $\{(x_1, y_0), \dots, (x_N, y_{N-1})\}$ be pairs generated by (2) (or equivalently (3)) where the matrix G is symmetric and $\|G\| \leq 1 - \kappa$. Let y^{extr} be the output of Algorithm 1. Consider the residual of the extrapolation

$$r^{\text{extr}} = g(y^{\text{extr}}) - y^{\text{extr}}.$$

Then, the norm $\|r^{\text{extr}}\|_W$ is bounded by

$$\|r^{\text{extr}}\|_W \leq \|I - (G - I)P\|_2 \frac{\xi^{i-1}}{1 + \xi^{2(i-1)}} \|r_1\|_W$$

where ξ is defined in (14) and W is a positive definite.

Proof. We start with the extrapolation step

$$y^{\text{extr}} = (Y - PR)\gamma_W.$$

Since for any vector v such that $\mathbf{1}^T v = 1$ we have

$$X^* v = (\sum_j v_j) x^* = x^*, \quad (37)$$

combining (37) with (7) gives

$$y^{\text{extr}} - x^* = ((G - I)^{-1} - P)R\gamma_W. \quad (38)$$

In addition, the residual r^{extr} can be written

$$r^{\text{extr}} = g(y^{\text{extr}}) - y^{\text{extr}} = (G - I)(y^{\text{extr}} - x^*). \quad (39)$$

Combining equations (38) and (39) gives

$$r^{\text{extr}} = (I - (G - I)P)R\gamma_W.$$

Its norm becomes

$$\begin{aligned} \|r^{\text{extr}}\|_W &= \|(I - (G - I)P)R\gamma_W\|_W \\ &\leq \|I - (G - I)P\|_2 \|R\gamma_W\|_W. \end{aligned}$$

By definition,

$$\gamma_W = \underset{\gamma}{\operatorname{argmin}} \|R\gamma\|_W \quad \text{subject to } \gamma^T \mathbf{1} = 1.$$

Using the structure of R , we have for some polynomial p_{γ_W} that $R\gamma_W = p(G)r_1$. This means

$$\|R\gamma_W\|_W = \min_{p: p \in \mathcal{P}_{N-1}, p(1)=1} \|p(G)r_1\|_W,$$

where \mathcal{P}_{N-1} is the space of polynomials of degree at most equal to one. Finally,

$$\begin{aligned} &\min_{p: p \in \mathcal{P}_{N-1}, p(1)=1} \|p(G)r_1\|_W \\ &\leq \|r_1\|_W \min_{p: p \in \mathcal{P}_{N-1}, p(1)=1} \|p(G)\|_2. \end{aligned}$$

The last term is bounded by (14), concluding the proof. \blacksquare

A.3 Proof of Proposition 4.2

Proposition A.3. *Consider the output of Algorithm 1 after N iterations of (3). If we can ensure*

$$y^{extr} = (Y - PR)\gamma_W = Y\tilde{\gamma} + X\gamma_{\tilde{W}},$$

then $y^{extr} - x^ = p_{N-1}(G)(y_0 - x^*)$, where p_N is a polynomial of degree $N - 1$ whose coefficients sum to one.*

Proof. First, consider $X\gamma_{\tilde{W}}$. We showed it can be written as a polynomial of degree 1, whose coefficients sum to one, with nonzero leading coefficient. Since $Y\tilde{\gamma}$ is, for any value of gamma, at most of degree $N - 1$, the whole expression still have a nonzero leading coefficient, and thus remains of degree N . Finally, because $\tilde{\gamma}$ sum to zero, we end with a polynomial of degree N , whose coefficients sum to one. ■

B Elements of Matrix Theory

B.1 Woodbury Matrix Identity

The Woodbury matrix identity [29] computes $(A + USV)^{-1}$ using a update on A^{-1} . The formula is

$$(A + USV)^{-1} = A^{-1} - A^{-1}U(S^{-1} + VA^{-1}U)^{-1}VA^{-1}. \quad (40)$$

B.2 Symmetric Solution of a System of Linear Equations

The results in [6, 13] show the symmetric solution of a system of equation. We give here the statement of Theorem 2 in [6].

Theorem B.1. *(Don, 1987) Consider the system of linear equations*

$$AX = B, \quad X = X^T.$$

Then, the system admits a symmetric solution if and only if

$$AA^{\sim}B = B \quad \text{and} \quad AB^T = BA^T.$$

If such condition is satisfied, the explicit formula for X reads

$$A^{\sim}B + (I - A^{\sim}A)(A^+B)^T + (I - A^{\sim}A)^T Z(I - A^{\sim}A),$$

where Z is an arbitrary symmetric matrix ($Z = 0$ gives the minimum norm solution) and A^{\sim} is a minimum-norm generalized reflexive pseudo-inverse of A , which means

$$AA^{\sim}A = A \quad , \quad A^{\sim}AA^{\sim} = A^{\sim} \quad \text{and} \quad A^{\sim}A = (A^{\sim}A)^T.$$

This theorem is useful when solving the optimization problem present in BFGS and DFP. As a corollary, we have that the solution of the transposed system

$$XA = B$$

is given by

$$\begin{aligned} X &= BA^+ + (BA^+)^T(I - AA^+) + (I - AA^+)^T Z(I - AA^+), \\ &= (BA^+)^T + (I - AA^+)^T BA^+ + (I - AA^+)^T Z(I - AA^+), \end{aligned} \quad (41)$$

under the condition that

$$BA^+A = B \quad \text{and} \quad A^T B = B^T A,$$

where A^+ is a generalized reflexive pseudo-inverse,

$$AA^+A = A \quad \text{and} \quad A^+AA^+ = A^+. \quad (42)$$

Here, the result is a bit more general, as A^+ does not need to be minimal-norm.

C Explicit Formulas for Quasi-Newton Methods

C.1 Generalized Quasi-Newton Step

Before analyzing each qN method, we introduce the generalized quasi-Newton step. We are looking for an approximation $H \approx (G - I)^{-1}$, where H satisfies the secant equation

$$HRC = YC.$$

After N iterations usual qN methods perform the step

$$y^{\text{extr}} = y_{N-1} - Hr_N.$$

Here, we are more interested in the *generalized quasi-Newton* (GqN) step

$$y_{\text{Generalized qN}} = (Y - HR)\gamma, \quad \forall \gamma^T \mathbf{1} = 1 \quad (43)$$

The next proposition shows that the choice of γ *does not change* y_{GqN} .

Proposition C.1. *Let H satisfies the secant equation (20). Then, the generalized qN step (22) is invariant in γ .*

Proof. Consider two different vectors γ_1 and γ_2 , whose entries sum to one. Then,

$$(Y - HR)\gamma_1 = (Y - HR)(\gamma_2 + (\gamma_1 - \gamma_2)) \quad (44)$$

Let $c = \gamma_1 - \gamma_2$. Since the entries of γ_1 and γ_2 sum to one, the entries of c sum to zero. In particular, because C is full column rank, this means there exist a vector v such that $Cv = c$. We can rewrite (44) into

$$\begin{aligned} (Y - HR)\gamma_1 &= (Y - HR)(\gamma_2 + Cv) \\ &= (Y - HR)\gamma_2 + (YC - HRC)v. \end{aligned}$$

Since H satisfies the secant equation (20), we have $(YC - HRC) = 0$ and

$$(Y - HR)\gamma_1 = (Y - HR)\gamma_2.$$

Because the choice of γ_1 and γ_2 was arbitrary, this prove the proposition. ■

In this section, we use this proposition to simplify the qN steps, in particular when $\gamma = \gamma_W$ (10) for some W . The previous results also hold when $H = J^{-1}$ and $JYC = RC$.

C.2 (Multi-Secant) Type-I Broyden

C.2.1 Generalized Multi-Secant Broyden Type-I Update

The multi-secant Broyden method solves

$$J = \underset{J}{\operatorname{argmin}} \|J - J_0\|_M \quad \text{s.t. } JYC = RC$$

First, we parametrize J to satisfy the constraints automatically. Let the matrix P_M such that

$$P_M YC = YC, \quad (I - P_M)YC = 0, \quad P_M M(I - P_M)^T = 0.$$

For example, we can take

$$P_M = YC ((YC)^T M^{-1} YC)^{-1} (YC)^T M^{-1}.$$

Now, without loss of generality, we divide J into

$$J = J_1 P_M + J_2 (I - P_M).$$

The optimization problem becomes

$$J = \underset{J=J_1 P_M + J_2 (I - P_M)}{\operatorname{argmin}} \|(J_1 - J_0)P_M + (J_2 - J_0)(I - P_M)\|_M \quad \text{s.t. } J_1 YC = RC$$

We can expand the objective into

$$\begin{aligned} & \|(J_1 - J_0)P_M + (J_2 - J_0)(I - P_M)\|_M \\ = & \operatorname{Trace} \left(M^{1/2} ((J_1 - J_0)P_M + (J_2 - J_0)(I - P_M)) M ((J_1 - J_0)P_M + (J_2 - J_0)(I - P_M))^T M^{1/2} \right) \end{aligned}$$

Using the property that $P_M M(I - P_M)^T = 0$,

$$\|(J_1 - J_0)P_M + (J_2 - J_0)(I - P_M)\|_M = \|(J_1 - J_0)P_M\|_M + \|(J_2 - J_0)(I - P_M)\|_M$$

Using the expression of P_M and the constraint $J_1 YC = RC$,

$$\begin{aligned} (J_1 - J_0)P_M &= (J_1 - J_0)YC ((YC)^T M^{-1} YC)^{-1} (YC)^T M^{-1} \\ &= (J_1 YC - J_0 YC) ((YC)^T M^{-1} YC)^{-1} (YC)^T M^{-1} \\ &= (RC - J_0 YC) ((YC)^T M^{-1} YC)^{-1} (YC)^T M^{-1} \end{aligned}$$

As this term is constant, we remove it from the optimization problem, which becomes

$$\begin{aligned} J &= \underset{J=J_1 P_M + J_2 (I - P_M)}{\operatorname{argmin}} \|(J_1 - J_0)P_M\|_M + \|(J_2 - J_0)(I - P_M)\|_M \quad \text{s.t. } J_1 YC = RC \\ &= \underset{J=J_1 P_M + J_2 (I - P_M)}{\operatorname{argmin}} \|(J_2 - J_0)(I - P_M)\|_M \end{aligned}$$

Clearly, the optimal solution is obtained when $J_2 = J_0$. Finally,

$$\begin{aligned} J &= J_1 P_M + J_2 (I - P_M) \\ &= RC ((YC)^T M YC)^{-1} (YC)^T M + J_0 (I - P_M) \\ &= J_0 + (RC - J_0 YC) ((YC)^T M^{-1} YC)^{-1} (YC)^T M^{-1} \end{aligned}$$

Now we use the Woodbury matrix identity (Appendix B.1),

$$\begin{aligned} J^{-1} &= J_0^{-1} - J_0^{-1} (RC - J_0 YC) + ((YC)^T M^{-1} YC + (YC)^T M^{-1} J_0^{-1} (RC - J_0 YC))^{-1} (YC)^T M^{-1} J_0^{-1} \\ &= J_0^{-1} + (YC - J_0^{-1} RC) + ((YC)^T M^{-1} J_0^{-1} RC)^{-1} (YC)^T M^{-1} J_0^{-1} \end{aligned}$$

This gives the generalized multi-secant update for the Type-I Broyden method.

Generalized Multi-Secant Type-I Broyden Matrix Update

$$J^{-1} = J_0^{-1} + (YC - J_0^{-1}RC) + ((YC)^T M^{-1} J_0^{-1} RC)^{-1} (YC)^T M^{-1} J_0^{-1}$$

Now, consider the generalized qN step (22)

$$y_{\text{Broyden Type-I}} = (Y - J^{-1}R)\gamma.$$

By Proposition A.3, this step is invariant in γ . In particular, if $\gamma = \gamma_W$ (10) with $W = (G - I)^{-1}M^{-1}J_0^{-1}$ (assuming this matrix symmetric positive definite) we can simplify the qN step. Indeed, using the formula (36) we have

$$\gamma_{(G-I)^{-1}M^{-1}J_0^{-1}} = \frac{(Y^T M^{-1} J_0^{-1} R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T M^{-1} J_0^{-1} R)^{-1} \mathbf{1}}.$$

Injecting those coefficients into the generalized qN step gives

$$(Y - J^{-1}R)\gamma_{(G-I)^{-1}M^{-1}J_0^{-1}} = (Y - J_0R)\gamma_{(G-I)^{-1}M^{-1}J_0^{-1}}.$$

Indeed, because by definition $C^T \mathbf{1} = 0$, the term highlighted in red in the equation below equal zero,

$$\begin{aligned} & J^{-1}R\gamma_{(G-I)^{-1}M^{-1}J_0^{-1}} \\ &= \left(J_0^{-1} + (YC - J_0^{-1}RC) + ((YC)^T M^{-1} J_0^{-1} RC)^{-1} (YC)^T M^{-1} J_0^{-1} R \right) \frac{(Y^T M^{-1} J_0^{-1} R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T M^{-1} J_0^{-1} R)^{-1} \mathbf{1}}. \end{aligned}$$

This step is a special instance of Algorithm 1 as shown in the following box.

Generalized Nonlinear Acceleration: Multi-Secant Type-I Broyden Step

- Set $W = (G - I)^{-1}M^{-1}J_0^{-1}$, compute γ_W with formula (36) (Assuming W symmetric positive definite).
- Set $P = J_0^{-1}$.

C.2.2 Simple Multi-Secant Broyden Type-I Update

In the standard case where $M^{-1} = I$ and $J_0^{-1} = \beta I$, it gives the following update.

Standard Multi-Secant Type-I Broyden Matrix Update

$$J^{-1} = \beta I + (YC - \beta IRC) + ((YC)^T RC)^{-1} (YC)^T$$

In this case, the call to Algorithm 1 is more straightforward.

Generalized Nonlinear Acceleration: Simple Multi-Secant Type-I Broyden Step

- Set $W = (G - I)^{-1}$, compute γ_W with proposition (2.1).
- Set $P = \beta I$, where β is a nonzero scalar.

As the structure of this GNA step matches Proposition 4.2, this instance is eligible for online acceleration.

C.3 (Multi-Secant) Type-II Broyden

C.3.1 Generalized Multi-Secant Broyden Type-II Update

The multi-secant Type-II Broyden update is similar to the Type-I. We start with

$$H = \underset{H}{\operatorname{argmin}} \|H - H_0\|_{M^{-1}} \quad \text{s.t. } HRC = YC$$

To be coherent with the units, we use M^{-1} as we estimate the inverse of $(G - I)$. Using the matrix

$$P_M = RC((RC)^T MRC)^{-1}(RC)^T M,$$

we decompose H into

$$H = H_1 P_M + H_2 (I - P_M).$$

The optimization problem becomes

$$H = \underset{H}{\operatorname{argmin}} \|(H_1 - H_0)P_M\|_{M^{-1}} + \|(H_2 - H_0)(I - P_M)\|_{M^{-1}} \quad \text{s.t. } HRC = YC$$

The left term is constant since

$$H_1 P_M = \underbrace{H_1 RC}_{=YC} ((RC)^T MRC)^{-1} (RC)^T M.$$

The optimization problem is thus optimal when $H_2 = H_0$. Finally, the update reads

$$H = H_1 P_M + H_2 (I - P_M) = H_0 + (RC - H_0 YC) ((RC)^T MRC)^{-1} (RC)^T M.$$

Generalized Multi-Secant Type-II Broyden Matrix Update

$$H = H_0 + (RC - H_0 YC) ((RC)^T MRC)^{-1} (RC)^T M.$$

Now, consider the generalized qN step (22). Using this update and $\gamma = \gamma_W$ with $W = M$ gives

$$y_{\text{Broyden Type-II}} = \left(Y - H_0 R + (RC - H_0 YC) ((RC)^T MRC)^{-1} (RC)^T MR \right) \frac{(R^T W R)^{-1} \mathbf{1}}{\mathbf{1}^T (R^T M R)^{-1} \mathbf{1}}$$

Since $C^T \mathbf{1} = 0$, the term in red simplifies and the updates reads

$$y_{\text{Broyden Type-II}} = (Y - H_0 R) \gamma_M.$$

We summarize in the next box the call to GNA to produce the same step than Broyden Type-II.

Generalized Nonlinear Acceleration: Generalized Multi-Secant Type-I Broyden Step

- Set $W = M$, compute γ_W with equation (10).
- Set $P = H_0$

C.3.2 Simple Multi-Secant Broyden Type-II Update

In the simple case, $M = I$ and $H_0 = \beta I$ where H_0 is a nonzero scalar. We summarize the simple Broyden update and its GNA call in the boxes below.

Generalized Multi-Secant Type-II Broyden Matrix Update

$$H = \beta I + (RC - \beta YC) ((RC)^T RC)^{-1} (RC)^T.$$

Generalized Nonlinear Acceleration: Simple Multi-Secant Type-II Broyden Step

- Set $W = I$, compute γ_W with equation (10).
- Set $P = \beta I$

C.4 (Multi-Secant) DFP

C.4.1 Generalized Multi-Secant DFP update

We start with

$$J = \underset{J}{\operatorname{argmin}} \|J - J_0\|_M \quad \text{s.t. } JYC = RC, \quad J = J^T.$$

We parametrize J as follow, using equation (41),

$$J = RC(YC)^+ + (RC(YC)^+)^T(I - YC(YC)^+) + (I - YC(YC)^+)^T Z(I - YC(YC)^+)$$

In particular, we choose as pseudo-inverse

$$(YC)^+ = (YC)_M^+ = ((YC)^T M^{-1}(YC))^{-1} (YC)^T M^{-1}.$$

Clearly we see that $(YC)_M^+$ satisfies (42). We also simplify the expression by writing

$$P_M = YC(YC)_M^+.$$

This way we ensure that J is symmetric and satisfies the secant equation. The optimization problem becomes

$$J = \underset{J}{\operatorname{argmin}} \|RC(YC)^+ + (RC(YC)^+)^T(I - P_M) + (I - P_M)^T Z(I - P_M) - J_0\|_M$$

First, we notice that

$$P_M M(I - P_M^T) = 0, \quad YC^+ = YC^+ P_M.$$

Using these properties and the definition of the weighted Frobenius norm (24) we have

$$J = \underset{J}{\operatorname{argmin}} \left\| (RC(YC)^+ - J_0) P_M \right\|_M + \left\| ((RC(YC)_M^+)^T + (I - P_M)^T Z - J_0) (I - P_M) \right\|_M$$

Using again the same ideas on the right term gives

$$J = \underset{J}{\operatorname{argmin}} \left\| (RC(YC)_M^+ - J_0) P_M \right\|_M + \left\| ((RC(YC)^+)^T P_M^T J_0) (I - P_M) \right\| + \left\| (I - P_M)^T (Z - J_0) (I - P_M) \right\|_W$$

As Z is the only free parameter, we have $Z = J_0$ and

$$J_{\text{DFP}} = RC(YC)_M^+ + (RC(YC)_M^+)^T(I - P_M) + (I - P_M)^T J_0(I - P_M) \quad (45)$$

To invert J we use two times the Woodbury matrix identity (Appendix B.1). First, consider the temporary matrix T

$$\begin{aligned} T &= (RC(YC)^+)^T + (I - P_M)^T J_0 \\ &= J_0 + W^{-1}(YC) ((YC)^T M^{-1}(YC))^{-1} ((RC) - J_0(YC))^T \end{aligned}$$

The approximation J becomes

$$\begin{aligned} J &= T + (RC(YC)_M^+ - T) P_M \\ &= T + (RC - TYC) ((YC)^T M^{-1}(RC))^{-1} (YC)^T M^{-1} \end{aligned}$$

Then, using the Woodbury matrix identity we have

$$\begin{aligned} J^{-1} &= T^{-1} - T^{-1}(RC - TYC) ((YC)^T M^{-1}YC + (YC)^T M^{-1}T^{-1}(RC - TYC))^{-1} (YC)^T M^{-1}T^{-1} \\ &= T^{-1} + (YC - T^{-1}RC) ((YC)^T M^{-1}T^{-1}RC)^{-1} (YC)^T M^{-1}T^{-1} \end{aligned}$$

It remains to compute T^{-1} using again the Woodbury matrix identity,

$$T^{-1} = J_0^{-1} + J_0^{-1}M^{-1}(YC) \left((RC)^T J_0^{-1}M^{-1}(YC) \right)^{-1} \left((YC) - J_0^{-1}(RC) \right)^T.$$

We summarize the update in the following box.

Generalized Multi-Secant DFP Matrix Update

$$J^{-1} = T^{-1} + (YC - T^{-1}RC) \left((YC)^T M^{-1}T^{-1}RC \right)^{-1} (YC)^T M^{-1}T^{-1}$$

where

$$T^{-1} = J_0^{-1} + J_0^{-1}M^{-1}(YC) \left((RC)^T J_0^{-1}M^{-1}(YC) \right)^{-1} \left((YC) - J_0^{-1}(RC) \right)^T.$$

C.4.2 Simple Multi-Secant DFP update

In the (much) simpler case where $M = (G - I)^{-1}$ and $J_0^{-1} = \beta I$, we have

$$\begin{aligned} T^{-1} &= \beta I + (RC) \left((RC)^T (RC) \right)^{-1} \left((YC) - \beta(RC) \right)^T, \\ J^{-1} &= T^{-1} + (YC - T^{-1}RC) \left((RC)^T T^{-1}RC \right)^{-1} (RC)^T T^{-1}. \end{aligned}$$

We can simplify the expression by looking at $(RC)^T T^{-1}$,

$$\begin{aligned} (RC)^T T^{-1} &= \beta(RC)^T + (RC)^T (RC) \left((RC)^T (RC) \right)^{-1} \left((YC) - \beta(RC) \right)^T \\ &= (YC)^T. \end{aligned}$$

The expression of J^{-1} becomes simpler,

$$J^{-1} = T^{-1} + (YC - T^{-1}RC) \left((YC)^T RC \right)^{-1} (YC)^T.$$

We can simplify even more since

$$\begin{aligned} T^{-1}RC &= \beta RC + (RC) \left((RC)^T (RC) \right)^{-1} \left((YC) - \beta(RC) \right)^T RC \\ &= RC \left((RC)^T (RC) \right)^{-1} (YC)^T RC. \end{aligned}$$

Multiplying the equation above by $\left((YC)^T RC \right)^{-1} (YC)^T$ gives

$$\begin{aligned} T^{-1}RC \left((YC)^T RC \right)^{-1} (YC)^T &= RC \left((RC)^T (RC) \right)^{-1} (YC)^T RC \left((YC)^T RC \right)^{-1} (YC)^T, \\ &= RC \left((RC)^T (RC) \right)^{-1} (YC)^T. \end{aligned}$$

We update the expression of J^{-1} using this result,

$$J^{-1} = T^{-1} + YC \left((YC)^T RC \right)^{-1} (YC)^T - RC \left((RC)^T (RC) \right)^{-1} (YC)^T.$$

If we replace T^{-1} in J^{-1} , we have

$$\begin{aligned} J^{-1} &= \beta I + (RC) \left((RC)^T (RC) \right)^{-1} \left((YC) - \beta(RC) \right)^T \\ &\quad + YC \left((YC)^T RC \right)^{-1} (YC)^T - RC \left((RC)^T (RC) \right)^{-1} (YC)^T, \\ &= \beta \left(I - (RC) \left((RC)^T (RC) \right)^{-1} (RC) \right) + YC \left((YC)^T RC \right)^{-1} (YC)^T \end{aligned}$$

This gives the standard DFP matrix update, summarized below.

Standard Multi-Secant DFP Matrix Update

$$J^{-1} = \beta \left(I - (RC) \left((RC)^T (RC) \right)^{-1} (RC)^T \right) + YC \left((YC)^T RC \right)^{-1} (YC)^T.$$

The generalized qN step (22) reads

$$y_{\text{DFP}} = (Y - J_{\text{DFP}}^{-1} R) \gamma$$

By proposition A.3, the choice of γ does not impact the result. We pick in particular

$$\gamma = \gamma_I = \frac{(R^T R)^{-1} \mathbf{1}}{\mathbf{1}^T (R^T R)^{-1} \mathbf{1}}.$$

Because $C^T \mathbf{1} = 0$, we have

$$(RC)^T R \gamma_I = C^T R^T \frac{(R^T R)^{-1} \mathbf{1}}{\mathbf{1}^T (R^T R)^{-1} \mathbf{1}} = 0.$$

Using this relation, the formula of the step becomes simpler,

$$y_{\text{DFP}} = Y \gamma_I - \left(\beta I + YC \left((YC)^T RC \right)^{-1} (YC)^T \right) R \gamma_I$$

We have a perfect match with the structure of Algorithm 1.

Generalized Nonlinear Acceleration: Multi-Secant DFP

- Set $W = I$ and compute γ_W using formula (10).
- Set $P = \beta I + YC \left((YC)^T RC \right)^{-1} (YC)^T$, where β is a nonzero scalar.

About the choice of W . We had two possible choices of W to simplify the expression of the qN step. For example, $W = (G - I)^{-1}$ leads to

$$\begin{aligned} J_{\text{DFP}}^{-1} R \gamma_{(G-I)^{-1}} &= \beta \left(I - (RC) \left((RC)^T (RC) \right)^{-1} (RC)^T \right) R \gamma_{(G-I)^{-1}}, \\ &= \beta R v, \quad v = \left(I - C \left((RC)^T (RC) \right)^{-1} (RC)^T (RC)^T \right) \gamma_I \end{aligned}$$

The major problem with this simplification is that it is unclear if the coefficient v_N , associated with R_N , is non-zero. This means we potentially lose the structure of (3) as we do not satisfy the assumptions of Proposition 4.2.

C.5 (Multi-Secant) BFGS

The Multi-Secant BFGS update follows the same reasoning that DFP method. It suffices to take the update of J_{DFP} (45), swap the matrices Y and R and replacing M^{-1} by M and this gives the multi-secant BFGS update, summarized in the box below.

Generalized Multi-Secant BFGS Matrix Update

$$H = YC(RC)_M^+ + (YC(RC)_M^+)^T(I - P_M) + (I - P_M)^T H_0(I - P_M)$$

where

$$\begin{aligned} (RC)_M^+ &= ((RC)^T M RC)^{-1} (RC)^T M \\ P_M &= RC(RC)_M^+ \end{aligned}$$

C.5.1 Simple Multi-Secant DFP update

Usually, the BFGS update is used with parameters $M = (G - I)^{-1}$ and $H_0 = \beta I$ where β is a nonzero scalar. We have the following simplifications.

$$\begin{aligned} (RC)_M^+ &= ((YC)^T RC)^{-1} (YC)^T, \\ P_M &= RC((YC)^T RC)^{-1} (YC)^T. \end{aligned}$$

In particular, the term $(YC(RC)_M^+)^T(I - P_M)$ is equal to zero as

$$\begin{aligned} (YC(RC)_M^+)^T(I - P_M) &= YC((YC)^T RC)^{-1} (YC)^T (I - RC((YC)^T RC)^{-1} (YC)^T), \\ &= 0. \end{aligned}$$

The update becomes

$$\begin{aligned} H &= YC(RC)_M^+ + \beta(I - P_M)^T(I - P_M), \\ &= \beta(I - P_M)^T + (YC - \beta RC)(RC)_M^+ \end{aligned}$$

Replacing $(RC)_M^+$ and P_M gives the "standard" multi-secant BFGS update.

Standard Multi-Secant BFGS Matrix Update

$$H = \beta \left(I - YC((YC)^T RC)^{-1} (RC)^T \right) + (YC - \beta RC)((YC)^T RC)^{-1} (YC)^T$$

This time, we use the parameters γ_W (10) in the generalized qN step (22) with $W = (G - I)^{-1}$. This simplifies the qN step since

$$(YC)^T R \gamma_{(G-I)^{-1}} = C^T Y^T R \frac{(Y^T R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T R)^{-1} \mathbf{1}} = 0.$$

Using this result, (22) becomes

$$(Y - H_{\text{BFGS}} R) \gamma_{(G-I)^{-1}} = \left(Y - \beta \left(I - YC((YC)^T RC)^{-1} (RC)^T \right) R \right) \gamma_{(G-I)^{-1}}.$$

Again, this matches perfectly the structure of Algorithm (1), whose parameters are summarized below.

Generalized Nonlinear Acceleration: Multi-Secant BFGS

- Set $W = (G - I)^{-1}$ and compute γ_W using Proposition (2.1).
- Set $P = \beta \left(I - YC((YC)^T RC)^{-1} (RC)^T \right)$, where β is a nonzero scalar.

C.6 SR-1 and SR- k

Here, we want to update the matrix H_0 with a symmetric matrix, with the lowest possible rank, so that the update H satisfies the secant equation (20). In other words, we want to solve

$$H = \underset{H}{\operatorname{argmin}} \mathbf{rank}(H - H_0), \quad HRC = YC, \quad (H - H_0) = (H - H_0)^T.$$

We can write the above problem as follow,

$$H = \underset{H}{\operatorname{argmin}} \mathbf{rank}(\Delta), \quad \Delta RC = YC - H_0RC, \quad \Delta = \Delta^T.$$

where $\Delta = H - H_0$. Using the solution to a symmetric system (41), we get

$$\Delta = (YC - H_0RC)(RC)^+ + ((YC - H_0RC)(RC)^+)^T(I - (RC)(RC)^+) + (I - (RC)(RC)^+)^T Z(I - (RC)(RC)^+),$$

Clearly, we can easily reduce the rank if $Z = 0$. It remains to find the right pseudo inverse $(RC)^+$ which minimizes the rank of

$$\Delta = (YC - H_0RC)(RC)^+ + ((YC - H_0RC)(RC)^+)^T(I - (RC)(RC)^+).$$

In particular, choosing

$$(RC)^+ = \left((YC - H_0RC)^T RC \right)^{-1} (YC - H_0RC)^T$$

leads to the symmetric rank k update

$$\Delta = (YC - H_0RC) \left((YC - H_0RC)^T RC \right)^{-1} (YC - H_0RC)^T.$$

This leads to the SR- k update below.

Symmetric Rank- k Update

$$H = H_0 + (YC - H_0RC) \left((YC - H_0RC)^T RC \right)^{-1} (YC - H_0RC)^T.$$

The sr- k step reads

$$y_{\text{sr-}k} = \left(Y - \left(H_0 + (YC - H_0RC) \left((YC - H_0RC)^T RC \right)^{-1} (YC - H_0RC)^T \right) R \right) \gamma.$$

Using

$$\gamma_{[(G-I)^{-1}+H_0]} = \frac{(R^T ((G-I)^{-1} + H_0) R)^{-1} \mathbf{1}}{\mathbf{1}^T (R^T ((G-I)^{-1} + H_0) R)^{-1} \mathbf{1}}$$

we have

$$HR\gamma_{[(G-I)^{-1}+H_0]} = H_0R\gamma_{[(G-I)^{-1}+H_0]}.$$

Notice that γ can be simplified using (36),

$$\gamma_{[(G-I)^{-1}+H_0]} = \frac{(Y^T R - R^T H_0 R)^{-1} \mathbf{1}}{\mathbf{1}^T (Y^T R - R^T H_0 R)^{-1} \mathbf{1}}.$$

The call to GNA is straightforward and summarized in the box below.

Generalized Nonlinear Acceleration: Symmetric Rank- k

- Set $W = (G - I)^{-1} + H_0$ and compute γ_W using (36).
- Set $P = H_0$ where H_0 is symmetric and $(G - I) \succ H_0$.

D Explicit Formulas for Krylov-subspace Methods

D.1 GMRES

In the case of GMRES (or equivalently MINRES, since we work with symmetric matrices), the iterations are *not* coupled. The algorithm creates a smart basis K_N of \mathcal{K}_N using Arnoldi (MINRES, [18]) or Lancos (GMRES, [21]) then computes

$$y_{\text{GMRES}} = \operatorname{argmin}_{x \in x_0 + \mathcal{K}_{N-1}} \|x - g(x)\|_2.$$

Here, we see that the iterates belongs to \mathcal{K}_{N-1} rather than \mathcal{K}_N .

In this section, we show that $y_{\text{GMRES}} = y_{\text{GNA}}$ when $W = I$ and $P = 0$. In particular, we show that

$$y_{\text{GMRES}} = x^* + p^*(G)(x_0 - x^*) \quad \text{and} \quad y_{\text{GNA}} = x^* + p^*(G)(x_0 - x^*), \quad \text{where } p^* = \operatorname{argmin}_{\substack{p:p(1)=1, \\ \deg(p) \leq N}} \|p(G)r_1\|_2. \quad (46)$$

D.1.1 GMRES and p^*

First, we start with the definition of the GMRES iterate. Indeed,

$$y_{\text{GMRES}} = \operatorname{argmin}_{x \in x_0 + \mathcal{K}_{N-1}} \|x - g(x)\|_2 = \operatorname{argmin}_{x \in x_0 + \mathcal{K}_{N-1}} \|(G - I)(x - x^*)\|_2$$

Using (31), we have

$$\begin{aligned} y_{\text{GMRES}} = x_0 + q^*(G)r_1, \quad q^* &= \operatorname{argmin}_{q:\deg(q) \leq N-2} \|(G - I)(x_0 + q(G)r_0 - x^*)\|_2 \\ &= \operatorname{argmin}_{q:\deg(q) \leq N-2} \|(I + (G - I)q(G))r_0\|_2 \end{aligned} \quad (47)$$

Instead of optimizing over all polynomial q of degree $\leq N - 2$, we optimize over p of degree at most $N - 1$ whose coefficients sum to one since

$$\min_{\substack{p=(I+(G-I)q(G)), \\ \deg(q) \leq N-1}} \|p(G)r_1\|_2 = \min_{\substack{p:p(1)=1, \\ \deg(p) \leq N}} \|p(G)r_1\|_2.$$

Let p^* be the optimal polynomial. We can deduce q^* from p^* using

$$q^*(G) = (G - I)^{-1}(p^*(G) - I)$$

If we replace q^* in (47) by the expression above, we have

$$\begin{aligned} y_{\text{GMRES}} &= x_0 + (G - I)^{-1}(p^*(G) - I)r_1, \\ &= x^* + (G - I)^{-1}p^*(G)r_1, \\ &= x^* + p^*(G)(G - I)^{-1}r_1, \\ &= x^* + p^*(G)(x_0 - x^*). \end{aligned}$$

This prove the first part of (46).

D.1.2 GNA and p^*

We start with the definition of the GNA iterate when $W = I$ and $P = 0$,

$$y_{\text{GNA}} = (Y - PR)\gamma_W = Y\gamma_I, \quad \gamma_I = \operatorname{argmin}_{\gamma:\gamma^T \mathbf{1}=1} \|R\gamma\|_2. \quad (48)$$

Since R is a basis of \mathcal{K}_N , we have that $R\gamma = p(G)r_1$. In addition, because r_i , the i -th column of R , can be written as

$$r_i = (G - I)(y_i - x^*) = (G - I)p_i(G)(x_0 - x^*) = p_i(G)r_1,$$

where $p_i(G)$ is defined in (3) and $p_i(1) = 1$, we have that for all $p : \deg(p) \leq N - 1$, $p(1) = 1$, there exist coefficients $\gamma : \gamma^T \mathbf{1} = 1$ such that

$$p(G) = \sum_{i=1}^N \gamma_i p_i(G) r_1 = \sum_{i=1}^N \gamma_i r_i = R\gamma.$$

By consequence,

$$R\gamma_I = p^*(G)r_1, \quad \gamma_I = \underset{\gamma: \gamma^T \mathbf{1}=1}{\operatorname{argmin}} \|R\gamma\|_2, \quad p^* = \underset{p: \deg(p) \leq N-1, p(1)=1}{\operatorname{argmin}} \|p(G)r_1\|_2$$

We can inject this solution in (48),

$$\begin{aligned} y_{\text{GNA}} &= Y\gamma_I, \\ &= x^* + (G - I)^{-1}R\gamma_I, \\ &= x^* + (G - I)^{-1}p^*(G)r_1, \\ &= x^* + p^*(G)(x_0 - x^*). \end{aligned}$$

This proves the second part of (46).

D.2 Conjugate Gradient

TBA

E Numerical Experiments

TBA