On Constrained Optimization by Adjoint based quasi-Newton Methods*

Andreas Griewank† and Andrea Walther†

August 29, 2002

Abstract

In this paper we propose a new approach to constrained optimization that is based on direct and adjoint vector-function evaluations in combination with secant updating. The main goal is the avoidance of constraint Jacobian evaluations and the reduction of the linear algebra cost per iteration to $\mathcal{O}(n+m)^2$ operations in the dense, unstructured case. A crucial building block is a transformation invariant two-sided-rank-one update (TR1) for approximations to the (active) constraint Jacobian. In this paper we elaborate its basic properties and report preliminary numerical results for the new total quasi-Newton approach on some small, equality constrained problems. A nullspace implementation under development is briefly described. The tasks of identifying active constraints, safe guarding convergence and many other important issues in constrained optimization are not addressed in detail.

Keywords Secant updating, quasi-Newton, KKT system, constrained optimization, Lagrangian gradient

1 Introduction

We begin by recapitulating a few well known facts from nonlinear optimization and automatic differentiation to motivate our approach and introduce our notation.

The KKT System

Once the active constraints are known, optima of a nonlinearly constrained optimization problem (NLP) are locally characterized as saddle points of the Lagrange function

$$\mathcal{L}(x,\lambda) \;\equiv\; f + \lambda^T c(x) \;\equiv\; f + \sum_{i=1}^{m} \lambda^{(i)} c_i(x).$$

Here the scalar function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is the objective, the vector function $c : \mathbb{R}^n \mapsto \mathbb{R}^m$ represents the $m \le n$ active constraints, and $\lambda = \left(\lambda^{(i)}\right)_{i=1\ldots m}$ denotes the corresponding vector of Lagrange multipliers. Thus the optimal values of the primal and dual variables $x \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}^m$ can be found by solving the stationarity condition

$$\begin{aligned} 0 \quad &= \quad \nabla_{x,\lambda}\mathcal{L}(x,\lambda) \equiv [g(x,\lambda), c(x)] \\ &\equiv \quad [\nabla f + \sum_{i=1}^{m} \lambda^{(i)} \nabla c_i(x), c(x)] \in \mathbb{R}^{n+m}. \end{aligned} \tag{1}$$

*Revised version of the Preprint *Towards Adjoint Based Constrained Optimization*, Preprint IOKOMO-08-2001, TU Dresden.

†Institute for Scientific Computing, Technical University Dresden, D-01062 Dresden, Germany

The Jacobian of this nonlinear system takes the partitioned form

$$\nabla^2_{x,\lambda}\mathcal{L}(x,\lambda) = \begin{bmatrix} \nabla^2 f + \sum_{i=1}^{m} \lambda^{(i)} \nabla^2 c_i(x) & \nabla c(x)^T \\ \nabla c(x) & 0 \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)} \quad . \qquad (2)$$

Being a Hessian at a saddle point this $(n+m) \times (n+m)$ matrix is always symmetric but neither positive nor negative definite. Only the reduced Hessian, i.e. the restriction of $\nabla^2_x \mathcal{L}(x,\lambda) = \nabla g(x,\lambda)$ to the nullspace of $\nabla c(x)$ can be expected to be positive definite in the vicinity of local minimizers.

Recently a considerable effort has gone into solving the KKT system by iterative solvers [16, 3]. For the most part these approaches work only if problem specific knowledge allows the provision of suitable (block) preconditioners with the symmetry of the system sometimes being ignored altogether. In this project we aim primarily at developing a general purpose solver for medium size problems involving "only" up to a few hundred variables.

Jacobian Evaluation Costs

All general purpose nonlinear programming algorithms that we aware of are based on the availability of the (active) constraint Jacobian $\nabla c(x_k) \in \mathbb{R}^{m\times n}$ at every major iterate $x_k \in \mathbb{R}^n$. For that purpose the Jacobian may be approximated by divided differences, evaluated using automatic differentiation tools, or coded by the user. Unless sparsity can be exploited, the cost incurred by divided differences or the forward mode of automatic differentiation is roughly equivalent to n extra evaluations of the constraint function $c(x)$. This must be considered a very significant computational cost.

When m is much smaller than n, the *reverse*, or *adjoint* mode of automatic differentiation offers some improvements as its cost grows only proportional to m rather than n. Hence the Jacobian $\nabla c(x)$ can always be obtained at a cost not exceeding a small multiple of $m = \min(m,n)$ times the cost of evaluating $c(x)$ by itself. We know of no other general bound on the relative cost of evaluating Jacobians and there are certainly no nontrivial lower bounds on that ratio. Amongst the reasons discussed in [6] is that, depending on the structure of the procedure for evaluating $c(x)$, there are very many different ways in which the chainrule can be applied to accumulate the Jacobian $\nabla c(x)$. Sometimes a *cross-country* application of this basic rule of calculus can yield Jacobians at a fraction of the cost required by the forward or reverse mode [11]. In principle any user with sufficient insight and patience can play the same combinatorial games in coding Jacobians by hand. In practice we suspect that mostly the forward mode with some local improvements will be employed. Anyway, the question of how best to evaluate complete Jacobians and Hessians remains a thorny issue and avoiding them altogether seems still a good idea.

Even if the constraint Jacobian can be computed quite cheaply at each new iterate the computation of a suitable representation for its nullspace requires a number of arithmetic operations of order mn^2 in the dense case. Overall, the approach advocated here probably makes computational sense in situations where the calculation of the Jacobian $\nabla c(x)$ and/or the computation of its null space is many times more expensive than evaluating $F \equiv [f, c]$. We believe this to be true for a very large class of practical problems.

Adjoint Evaluation Costs

The required human and run-time costs for the Jacobian may seem justified to some algorithm designers, because it already appears in the combined gradient

$$\nabla \mathcal{L}(x,\lambda) \equiv [g(x,\lambda), c(x)] \quad ,$$

whose accurate evaluation is certainly a prerequisite for computing its roots with any accuracy.

However, as is still not widely appreciated, the occurrence of the Jacobian $\nabla c(x)$ in the definition of

$$g(x,\lambda) = \nabla f + \nabla c(x)^T \lambda$$

does not imply at all that it must be evaluated in its entirety as an $m \times n$ matrix. Instead one may call on the reverse mode of automatic differentiation to evaluate only the vector $g(x,\lambda)$ as adjoint of $F(x)$ at a small multiple of the cost of evaluating the objective and constraints by themselves. Thus we pursue the aim of limiting the function evaluation effort per major iteration to a couple of evaluations of $F(x)$ and $g(x,\lambda)$. At the same time we wish to limit the linear algebra cost per step to $\mathcal{O}(n+m)^2$ or even $\mathcal{O}(k(n+m))$ where k is the number of steps taken so far.

2 The total quasi-Newton Approach

The design goals mentioned above strongly suggest that we should approximate the KKT Jacobian $\nabla^2 \mathcal{L}$ by secant updating. Hence, we wish to sequentially generate symmetric matrices $B_k \in \mathbb{R}^{n \times n}$ approximating the Hessians $\nabla_x^2 \mathcal{L}(x_k, \lambda_k) = \nabla g(x_k, \lambda_k)$ and rectangular matrices $A_k \in \mathbb{R}^{m \times n}$ approximating the constraint Jacobians $\nabla c(x_k)$, at a sequence of iterates (x_k, λ_k). We will simply refer to the B_k and the A_k as approximating Hessians and Jacobians, respectively. The notion of making do with approximate Jacobians is not even broached in the recent text books [10] and [12]. We use the adjective *total* rather than the less dramatic *full* for our approach because the latter is sometimes used to describe secant updates of B_k rather than its projection.

Suppose the initializations B_0 and A_0 are chosen such that for $k = 0$ the linear system

$$\begin{bmatrix} B_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} s_k \\ \sigma_k \end{bmatrix} = - \begin{bmatrix} g_k \\ c_k \end{bmatrix} \tag{3}$$

is easily solvable. Then, and if all corrections $B_{k+1} - B_k$ and $A_{k+1} - A_k$ are of low rank, the same will be true for $k > 0$ as one may employ a structured variant of the Sherman-Morrison-Woodbury formula [5]. A nullspace variant currently under development is sketched in Section 5.

We will try to get away with a rank three update overall and keep in mind that for really large problems a limited memory version that stores and manipulates only a fixed number of update vectors should be developed as well. In view of discretizations of operator equations one should also be aware that grid invariant superlinear convergence can only be achieved if in the limiting Hilbert space setting the discrepances between the exact Frechet derivatives $\nabla_x^2 \mathcal{L}(x_*, \lambda_*)$ and $\nabla c(x_*)$ at the solution and their initial approximations B_0 and A_0 are relatively compact operators [8].

Approximating the Hessian

The task of updating B_k or a suitable projection/restriction has received considerable attention in the optimization literature, and some of the established techniques should

be applicable here too. However, since in our scenario the constraint Jacobian is also approximated, we can no longer expect that the final approach is essentially tangential to the feasible set $c^{-1}(0)$. In other words, even during the final approach to a local minimum the steps $s_k = x_{k+1} - x_k$ may have strong normal components, and thus feasibility may not be approached much faster than optimality. This means in particular that there is no need for the curvature $s_k^T \nabla_x \mathcal{L}(x_k, \lambda_k) s_k$ to ever be positive. Hence Powell-like update conditions [13] may never be satisfied. In [15] various strategies for updating approximations to one-sided projections of the Hessian over not predominantly tangential steps were analyzed – with somewhat inconclusive results.

Consequently, in our scenario a good choice might be the damped symmetric-rank-one (SR1) update

$$B_{k+1} = B_k + \beta_k \frac{(w_k - B_k s_k)(w_k - B_k s_k)^T}{(w_k - B_k s_k)^T s_k} \quad , \tag{4}$$

which is designed to satisfy for $\beta_k = 1$ the secant condition

$$B_{k+1} s_k \;=\; w_k \equiv g(x_{k+1}, \lambda_k) - g(x_k, \lambda_k) \in \mathbb{R}^n \quad , \tag{5}$$

where x_k and λ_k denote the current point and Lagrange multipliers. The two vectors on the right hand side can be evaluated as part of the optimization procedure without — as we would like to stress again — the need to ever evaluate the full constraint Jacobian $\nabla c(x)$.

Toint [4] has used the SR1 formula with good success for updating approximations to element Hessians of partially separable objective functions. As also suggested in [12] he determined the scaling factor β_k using a test that involves norms of the vectors occurring in the updating formula. At least aesthetically, this contingency measure is not very pleasing because it destroys any prospect of maintaining invariance with respect to linear transformations on the variable domain. In Section 5 we will sketch an implementation where β_k is selected such that the projection of B_k onto the nullspace of A_k stays positive definite.

Of course there is a large variety of other issues to be resolved but we will concentrate in this first proposal of the novel approach on the challenging task of designing a suitable secant formula for the constraint Jacobian.

Approximating the Jacobian

In contrast with the theoretical and practical attractions of the BFGS formula for positive definite Hessians, secant updates for nonsymmetric Jacobians have rarely met with success across a wide range of problems. One possible explanation is that least change updates like the good and the bad Broyden formula are strongly dependent on inner product norms and hence the scaling in the domain or range of the underlying vector function. In contrast the BFGS and all other updates of the Broyden class including SR1 formula are known to be invariant with respect to linear transformations on the variable domain, provided the initial B_0 is adjusted accordingly.

The key ingredient that allows us to achieve (linear) transformation invariance here is the imposition of the *adjoint*, or *transposed*, secant condition

$$\sigma_k^T A_{k+1} \;=\; \mu_k^T \;\equiv\; \sigma_k^T \nabla c(x_{k+1}) \quad \text{with} \quad \sigma_k \;\equiv\; \lambda_{k+1} - \lambda_k \in \mathbb{R}^m \quad . \tag{6}$$

Here μ_k can be calculated as

$$\mu_k \;\equiv\; g(x_{k+1}, \lambda_{k+1}) - g(x_{k+1}, \lambda_k) \in \mathbb{R}^n \quad . \tag{7}$$

In other words we assume that our optimization procedure generates distinct Lagrange multiplier vectors λ_k and λ_{k+1} and involves the corresponding adjoint values $g_{k+1} \equiv g(x_{k+1}, \lambda_{k+1})$ and $g(x_{k+1}, \lambda_k)$ at the new iterate x_{k+1} anyway.

Naturally, we will also impose the classical, or *direct* secant condition

$$A_{k+1} s_k \;=\; y_k \equiv c(x_{k+1}) - c(x_k) \quad \text{with} \quad s_k \equiv x_{k+1} - x_k \in \mathbb{R}^n \quad . \tag{8}$$

Unless the constraint function $c(x)$ is affine the two conditions will in general not be exactly consistent. More specifically, the multiplication of (6) by s_k from the right and (8) by σ_k^T from the left, respectively, yield the following alternative expressions for $\sigma_k^T A_{k+1} s_k$

$$\mu_k^T s_k = \sigma_k^T \left[\nabla c(x_{k+1})\right] s_k \;\not\equiv\; \sigma_k^T \left[\int_0^1 \nabla c(x_k + \alpha\, s_k) d\,\alpha\right] s_k \equiv \sigma_k^T y_k \quad . \tag{9}$$

Obviously, the deviation between the two matrices in the middle and hence that between the two dot products is only of order $\mathcal{O}(\|\sigma_k\|\,\|s_k\|^2)$. Here we have tacitly assumed that the Jacobian $\nabla c(x)$ is at least Lipschitz continuous as we will do throughout the paper. This level of discrepancy between approximating and exact Jacobians is inherent in the quasi-Newton updating philosophy, which replaces tangents with secants over increments of size $\|s_k\|$.

3 The TR1 update

It can be easily seen that the rank-one formula

$$A_{k+1} \equiv A_k + \frac{(y_k - A_k s_k)(\mu_k^T - \sigma_k^T A_k)}{\mu_k^T s_k - \sigma_k^T A_k s_k} \tag{10}$$

satisfies the direct secant condition exactly and the adjoint version up to $\mathcal{O}(\|\sigma_k\|\,\|s_k\|)$. These roles are reversed if one replaces the dot product $\mu_k^T s_k$ in the denominator by $\sigma_k^T y_k$, a slight change that should not effect the quality of the update significantly. Irrespective of the slight arbitrariness in the denominator we will refer to (10) as the two-sided-rank-one (TR1) update. It generalizes the classical SR1, to which it reduces when the Jacobian $\nabla c(x)$ is in fact a symmetric Hessian because $c(x)$ happens to be a gradient. Not surprisingly, TR1 shares many desirable and some not so desirable properties that are familiar from SR1.

Heredity and Termination

In contrast to the BFGS formula, SR1 is *hereditary* on affine gradients in that it maintains the validity of all previous secant conditions and thus yields the exact Hessian in at most n linearly independent but otherwise arbitrary, steps. Similarly, we find here that in the affine case $c(x) = c(0) + A_* x$ the TR1 update is unique and the discrepancy matrix $D_k \equiv A_k - A_*$ satisfies the recurrence

$$D_{k+1} \equiv D_k - \frac{D_k s_k \sigma_k^T D_k}{\sigma_k^T D_k s_k} \tag{11}$$

provided the denominator $\sigma_k^T D_k s_k$ does not vanish. Any nullvector of D_k or its transpose will also be a nullvector of D_{k+1} or its transpose, respectively. Thus the update is indeed hereditary.

Moreover, provided $\sigma_k^T D_k s_k \neq 0$, s_k and σ_k become new null-vectors of D_{k+1} and its transpose, respectively. Consequently the rank of D_k must be reduced by one and after at most $m = \min(m, n)$ such updates the discrepancy vanishes exactly. This property is at least theoretically very interesting, though even on medium size problems one often does not wish to take $\Theta(m)$ steps.

Transformation Invariance

Like SR1 the more general TR1 update has apparently no least change characterization in terms of any particular matrix norm. However, one should note that the rank of D_k is a transformation invariant measure of discrepancy.

Let us be optimistic and suppose that from a given triplet (x_0, B_0, A_0) and $\lambda_0 = 0$ our method as defined by the relations (3), (4), and (10) yields a uniquely determined infinite sequence of points (x_k, λ_k) and matrix pairs (B_k, A_k). In other words, we simply exclude the contingency of zero denominators and assume that $f(x)$ and $c(x)$ are well defined wherever they are to be evaluated.

Now let C and R be any nonsingular square matrices of size n and m, respectively. Then the transformed problem of optimizing $\tilde{f}(\tilde{x}) \equiv f(C\tilde{x})$ subject to $0 = \tilde{c}\,(\tilde{x}) \equiv R\,c\,(C\,\tilde{x})$ is largely equivalent to the original optimization problem, which is obtained when C and R are the identity matrix of their respective dimension. Then one can check by induction on k that the proposed method applied to the transformed problem maintains the relations

$$x_k = C\,\tilde{x}_k, \quad \tilde{\lambda}_k^T R = \lambda_k^T, \quad \tilde{B}_k = C^T B_k C, \quad \tilde{A}_k = R\,A_k C \tag{12}$$

provided they are true initially for $k = 0$. This is a very desirable feature even though the appropriate initialization of B_0 and A_0 may not be obvious at all. Not surprisingly, as we will see in the next section for A_0, these initializations may be crucial for the performance of an algorithm.

Blow-up and Special Steps

As in the case of SR1 a downside associated with heredity is that the TR1 update may blow up due to the denominator being small or even zero. This happens in particular if one of the discrepancies $(y_k - A_k s_k)$ or $(\mu_k - A_k^T \sigma_k)$ vanishes exactly. If both happen to vanish simultaneously, one could simply skip updating for the current step.

For a full quasi-Newton step (s_k, σ_k) on the KKT system as defined by (3) we have $A_k\,s_k = -c(x_k)$. Thus $y_k = A_k s_k$ would imply exact primal feasibility at the new point $x_{k+1} = x_k + s_k$ in that $c(x_{k+1}) = 0$. In principle this is a rather desirable effect but as there is no reason why $(\mu_k - A_k^T \sigma_k)$ should vanish simultaneously we must then come up with another primal step $\tilde{s}_k \neq 0$ for the update. This could be any direction $\tilde{s}_k$ for which $A_k \tilde{s}_k$ does not agree with $\nabla c(x_{k+1})\tilde{s}_k$. This directional derivative can easily be evaluated in the forward mode of automatic differentiation or possibly approximated by divided differences. Such special updating steps were already utilized by M.J.D. Powell [14] in the middle of the last century.

If on the other hand $(\mu_k - A_k^T \sigma_k)$ vanishes or is very small but $(y_k - A_k s_k)$ is sizeable then we have to play the game the other way round by picking special adjoint steps $\tilde{\sigma}_k$ with $\tilde{\sigma}_k^T A_k \neq \tilde{\sigma}_k^T \nabla c(x_{k+1})$. The last vector can be evaluated in the reverse mode of automatic differentiation, but not approximated by differences at a reasonable cost. Since we have for a full quasi-Newton step $B_k s_k + A_k^T \sigma_k = -g(x_k, \lambda_k)$ one finds after some elementary rearrangements that

$$g(x_{k+1}, \lambda_{k+1}) = (w_k - B_k s_k) + (\mu_k - A_k^T \sigma_k) \quad . \tag{13}$$

Hence we observe that when the direct secant condition (5) for B_{k+1} and the adjoint secant condition (6) for A_{k+1} are already satisfied by B_k and A_k then we must have dual feasibility at the new point in that $g(x_{k+1}, \lambda_{k+1}) = 0$.

In summary, we observe the usual quasi-Newton dichotomy, namely that at each major iteration significant progress is being made either in stepping towards the solution vector or in updating derivative matrices. Obviously much remains to be investigated, especially with regards to damping factors for stepping and updating.

4 Numerical Feasibility Test

To check whether the idea of approximating the constraint Jacobian by the proposed twosided update TR1 has any merit we conducted two kinds of numerical experiments. Both tests have preliminary character and were carried out in MATLAB. Firstly, to investigate the properties of TR1 by itself we look at the special situation where $m = n$. In such nonsingular *square* cases the nonlinear system $c(x) = 0$ has locally just one feasible point x_*. However, in addition to computing the solution x_* itself, an optimization method will simultaneously solve the adjoint equation

$$0 = g(x, \lambda) \equiv \nabla f(x) + \nabla c(x)^T \lambda \quad ,$$

which is of course linear in the multiplier vector λ. The solution vector λ_* might then be used to compute the reduced derivative of the feasible value $f(x_*)$ with respect to other problem parameters when they are considered variable after all. More generally, one might interpret Lagrange multipliers and other intermediate adjoint variables as impact factors with regards to the given target function $f(x)$. These goal-oriented sensitivities can be used to estimate errors from various sources and to ameliorate their influence on target values by suitable modification, e.g. grid adaption [1].

When the square Jacobians $\nabla c(x_k)$ are exactly known the objective function $f(x)$ and its gradient have no impact on the Newton steps $s_k = -\nabla c(x_k)^{-1} c(x_k)$. This will certainly be true on the final approach to x_*, while earlier on the objective function might conceivably cause a reduction in stepsize or a contraction of the trust region radius. Possibly still a bit later the same effect is likely to occur when $c(x) = 0$ is solved by classical secant methods like the good or bad Broyden formula.

In contrast the proposed TR1 update directly involves adjoint information and is therefore strongly dependent on the objective function f via its gradient ∇f. In a very optimistic mood one might hope that this weighting of the primal solution components will guide the iteration such that the function values $f(x_k)$ converge faster than the error norm $\|x_k - x_*\|$ towards their limit $f(x_*)$.

Since the following results were obtained with MATLAB we plot iteration numbers rather than timing results. Also, the problem considered is rather special, namely a central difference discretization of the second order ODE

$$\frac{d^2}{dt^2} x(t) + \delta \frac{d}{dt} x(t) + \gamma \exp(x(t)) = 0 \quad \text{for} \quad 0 < t < 1 \tag{14}$$

with homogeneous Dirichlet boundary conditions $x(0) = 0 = x(1)$. It is well known that for sufficiently small δ and γ this problem is elliptic and has therefore a unique solution. If γ is increased the problem becomes more difficult and reaches a fold point at a certain critical value $\gamma_*(\delta)$. The numerical results displayed below were obtained for the subcritical parameter values $\delta = 1$ and $\gamma = 0.5$ or $\gamma = 2.0$.

Throughout we used objective functions of the form

$$f(x) = \int_0^1 \omega(t) x(t) dt$$

and their discretization by the trapezoidal rule. The weighting function $\omega(t)$ was set constantly equal to 1 for the calculations reported in Figures 1 and 2. For the calculation in Figure 3 we used an f_1 defined by $\omega_1(t) = 8\,t(1-t)$ and f_2 defined by $\omega_2(t) = 2 - 7.6\,t(1-t)$. All iterations were terminated when the max norm of the constraints had been reduced below 10^{-10}.

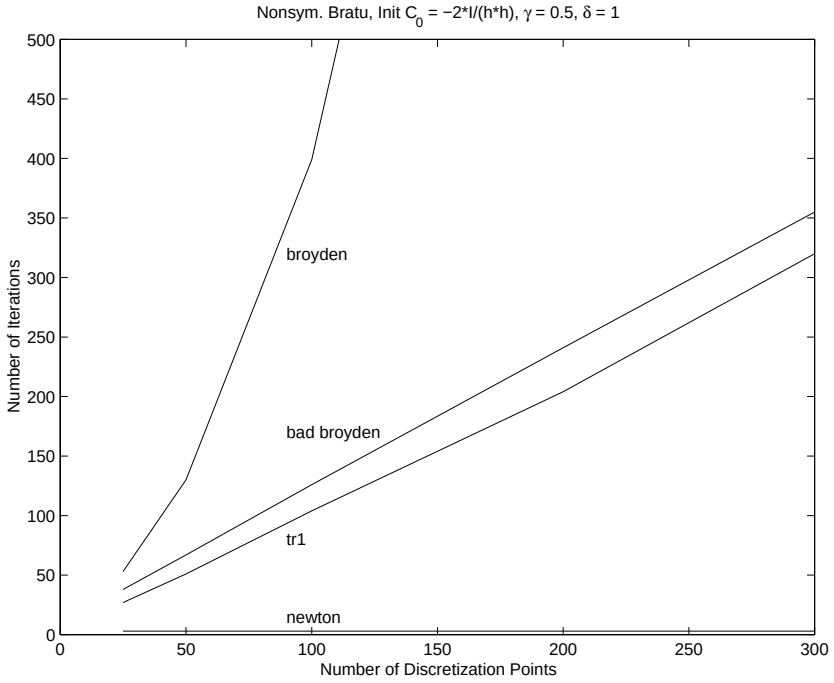

Figure 1: Iterations for Newton and quasi-Newton from "simple" Initialization

The unit interval was uniformly subdivided into $n+1$ intervals of length $h = 1/(n+1)$. All calculations were started from the origin $(x, \lambda) = 0 \in \mathbb{R}^{2n}$. The approximating Jacobian A_0 was initialized either as the identity scaled by $-2/h^2$ or the tridiagonal matrix with $-2/h^2$ in the diagonal and $1/h^2$ in the super- and sub-diagonal. For the first *simple* choice the discrepancy $I - A_0^{-1}\nabla c(x_*)$ acquires more and more large singular values as n grows, whereas for the second *smart* choice, the number of singular values above any particular threshold stays bounded irrespective of n. In terms of functional analysis this means that the differential operator on the left of (14) preconditioned by the inverse of its leading term x_{tt} is a compact perturbation of the identity. Under these circumstances it can be shown [8] that Broydens *good* method achieves superlinear convergence if started from the smart initialization but at best linear convergence if started from the simple choice. Correspondingly one would expect grid invariant iteration numbers to reach certain tolerances with the smart initialization but a strong increase in over the iterations count with the simple choice. A similar behavior is likely for other low rank updating methods though this has not been established as far as we know.

As one can see in Figure 1 we get convergence of all four methods, i.e. Newton, TR1, good Broyden and bad Broyden. In all cases full steps were taken throughout and the simple initial $A_0 = -2I/h^2$ was used. The TR1 update was used in the form (10). Replacing $\mu_k^T s_k$ by $\sigma_k^T y_k$ made no noticeable difference for the number of steps. While Newton exhibits a grid invariant convergence behavior, the three quasi-Newton

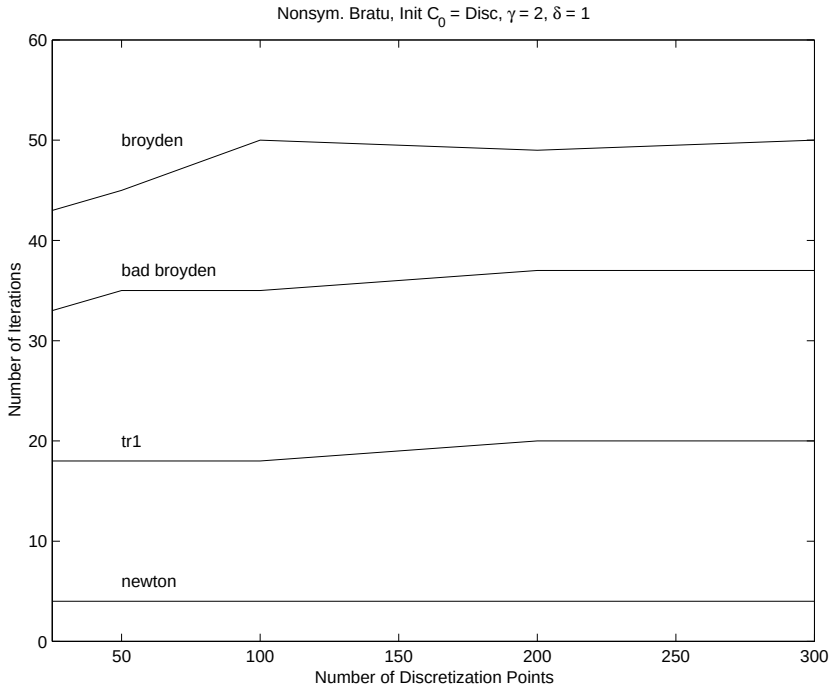

Figure 2: Iterations for Newton and quasi-Newton from "smart" Initialization

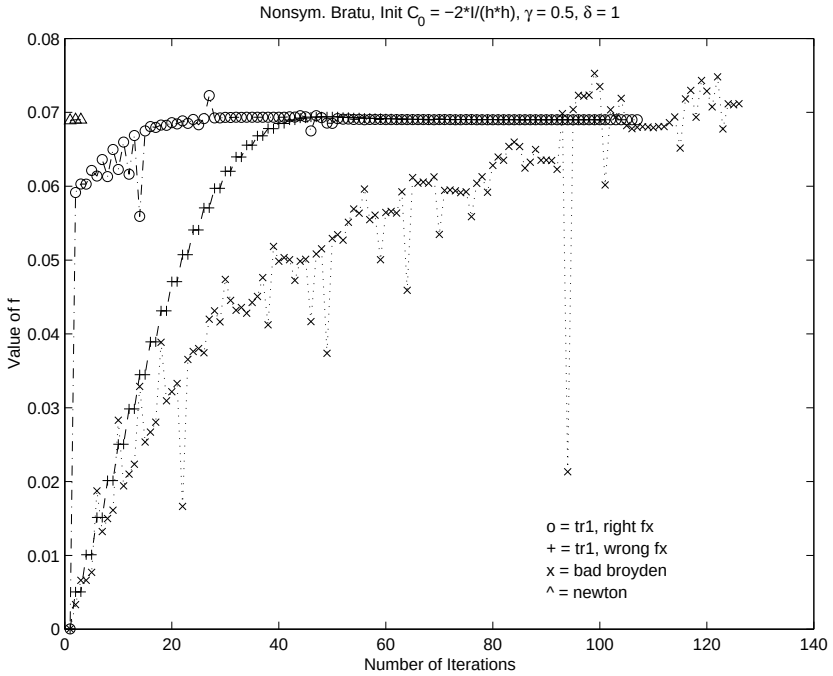

Figure 3: Function Value History for Newton, Broyden, and TR1

methods need at least a number of steps proportional to discretized space dimension. The supposedly bad Broyden update performs much better than the standard "good" version and both are slower than our favorite TR1.

As one can see in Figure 2 the three quasi-Newton methods become essentially grid invariant like Newton's method when A_0 is initialized smartly to the tridiagonal discretization of x_{tt} but their relative performance ranking stays the same. As one can see in Figure 3 solving the right adjoint system in tandem does actually improve significantly the convergence speed of the function values. The triangles represent the values attained by Newton's method, which reaches the feasible solution x_* and the corresponding function value $f_1(x_*)$ as defined above in just a few steps. The circles represent the values of f_1 attained by the TR1 scheme when the adjoint is defined by the correct gradient $\nabla f_1(x)$. The plus signs depict the f_1 values attained by the TR1 scheme when the adjoint iteration solves the system $\nabla c(x)^T\lambda = -\nabla f_2(x)$ corresponding to the other objective function. Hence, at least on this particular example there is some support for the optimism that the TR1 update guides the iterates in such a way that they converge particularly fast in those components that strongly impact the given objective function.

Equality Constrained Optimization

We tested the combination of TR1 for the approximating Jacobians A_k and SR1 for the approximating Hessians B_k on the set of small problems used originally by Wagner and Todd [15]. The results are listed in Table 1 with n and m the number or variables and constraints, respectively. The third and forth column were taken from [15] and represent the number of steps required by a full Newton implementation and a mixed quasi-Newton method based on the DFP related Hessian-update proposed in [15]. Here *mixed* means that the Hessian is approximated by secant updating but the Jacobian is exactly evaluated, as is currently customary in constrained optimization.

Table 1: Iteration Counts for Newton, Mixed, and Total quasi-Newton Methods

	n	m	Exact	W & T/G	STR1
Wright	2	1	5	6	10
Himmelblau	3	2	3	5	9
Powell	5	3	5	6	15
Wright	5	3	**	13	9
HS100	7	2	11	13	19
HS111	10	3	43	23	**
HS104	8	4	5	8	**
Bryd	2	1	9	7	10
Bryd	2	1	8	7	9
Yuan	2	1	6	9	9
Yuan	2	1	5	7	7
HS39	4	2	4	10	**
HS93	6	2	4	18	**

The extra column labeled STR1 represents the number of iterations for our SR1–TR1 combination, started with a finite difference initialization of both Jacobian and Hessian. As in [15] no update damping or step stabilization strategy was applied, which explains in particular the convergence failures on problems HS111, HS104, HS39, and HS93. The denominator of the TR1 formula was defined as either $\mu_k s_k$ or $\sigma_k^T y_k$, whichever had the larger absolute value.

As one can see the number of iterations for the total quasi-Newton approach is on some problems similar to and on some problems about twice as large as that for the mixed quasi-Newton approach. With the exception of the problem "Wright" the new approach requires a few more steps then the schemes based on exact first and possibly second derivative matrices. This effect was to be expected and is certainly acceptable assuming each iteration is significantly cheaper for the total quasi-Newton approach. To see that this is a realistic possibility we briefly sketch a null-space implementation with quadratic complexity in the next section.

5 A possible Nullspace Implementation

Rather than dealing with the approximating Jacobian and Hessian A_k and B_k explicitly we suggest storing and manipulating the five matrices

$$Y_k, C_k \in \mathbb{R}^{m \times n} \ , \quad L_k \in \mathbb{R}^{m \times m} \ , \quad Z_k \in \mathbb{R}^{(n-m) \times n} \ , \quad R_k \in \mathbb{R}^{(n-m) \times (n-m)}$$

such that Y_k and Z_k have orthonormal rows, L_k and R_k are lower triangular and

$$L_k Y_k = A_k \ , \quad Y_k Z_k^T = 0 \ , \quad R_k^T R_k = Z_k B_k Z^T \ , \quad C_k = Y_k B_k \quad .$$

The total storage requirement is $m^2 + 3n^2/2$, which would be reduced by $m^2/2$ if $C_k = Y_k B_k$ was split into $Y_k B_k Z_k^T$ and $Y_k B_k Y_k^T$ with the latter stored in symmetric mode.

Due to the orthogonality of $\left[Y_k^T, Z_k^T\right]$ we have $Y_k^T Y_k + Z_k^T Z_k = I \in \mathbb{R}^{n \times n}$ so that for any vector $v \in \mathbb{R}^n$

$$B_k \, v = Y_k^T C_k \, v + Z_k^T \left(R_k^T R_k Z_k \, v + Z_k C_k^T Y_k \, v \right) \quad .$$

This computation costs $3n^2 + mn + m^2$ operations and is thus only m^2 multiplications less expensive than solving the KKT linearization (3) in the form

$$s_k = Y_k^T \tilde{c}_k - Z_k^T R_k^{-1} R_k^{-T} Z_k \left(g_k + C_k^T \tilde{c}_k \right)$$

where $\tilde{c}_k = -L_k^{-1} c_k$. The corresponding Lagrange multiplier step is simply

$$\sigma_k = -L_k^{-T} \left(C_k s_k + Y_k g_k \right) \quad .$$

Updating Procedure

It can be easily checked that the determinant of the whole KKT matrix is equal to $[\det(R_k) \det(L_k)]^2$ so that its conditioning can be partially controlled by the updates for R_k and L_k. Using well established techniques from linear algebra [2] we can update the factorized form $A_k = L_k Y_k$ to

$$A_{k+1} = L_{k+1} Y_{k+1} = A_k + r_k \rho_k^T / \delta_k \quad ,$$

where

$$r_k = y_k - A_k \, s_k \ , \quad \rho_k^T = \mu_k^T - \sigma_k^T \, A_k$$

and

$$\sigma_k^T r_k \approx \delta_k \approx \rho_k^T \sigma_k \quad .$$

Correspondingly we may update Z_k by the Sherman-Morrison-Woodbury like formula

$$Z_{k+1} = Z_k \left[I - \rho_k b_k^T / (\delta_k + \rho_k^T b_k) \right]$$

where for some α_k determined by a certain quadratic equation

$$b_k = Y_k^T L_k^{-1} r_k + \alpha_k Z_k^T Z_k \rho_k \quad .$$

The projected Hessian $Z_k^T B_k Z_k = R_k^T R_k$ must be updated twice, first with $\beta_{k+\frac{1}{2}} \geq \|b_k\|^2$ to the intermediate version

$$\begin{aligned} R_{k+\frac{1}{2}}^T R_{k+\frac{1}{2}} &= Z_{k+1} B_k Z_{k+1}^T \\ &= R_k^T R_k - \frac{Z_k b_k \rho_k^T Z_k^T + Z_k \rho_k b_k^T Z_k^T}{(\delta_k + \rho_k^T b_k)} + \beta_{k+\frac{1}{2}} \frac{Z_k \rho_k \rho_k^T Z_k^T}{(\delta_k + \rho_k^T \delta_k)^2} \quad . \end{aligned}$$

For this rank-two modification we may utilize well-established techniques for updating Cholesky factorization of positive definite Hessians in unconstrained and constrained optimization [12]. To maintain the semi-definiteness implicit in the Cholesky factorization we may have to choose $\beta_{k+\frac{1}{2}}$ bigger than its natural value $\|b_k\|^2$.

Subsequently, we may apply the SR1 update (4) in the form

$$R_{k+1}^T R_{k+1} = R_{k+\frac{1}{2}}^T R_{k+\frac{1}{2}} + \beta_k Z_{k+1} z_k z_k^T Z_{k+1}^T / z_k^T s_k$$

where $z_k = (w_k - B_k s_k)$ can be computed after a full quasi-Newton step as $(g_{k+1} + Y_k^T L_k^T \sigma_k)$. Here β_k may need to be chosen smaller than its natural value 1 to maintain positiv definiteness. In any case the composite transition from $R_k^T R_k$ to $R_{k+1}^T R_{k+1}$ is a rank-three update as was to be expected.

The update of the cross term $C_k = Y_k B_k$ also proceeds in two stages. Firstly we obtain $C_{k+\frac{1}{2}}$ by applying to C_k all the modifications that Y_k undergoes in its update to Y_{k+1}. For that purpose we are currently using standard Givens rotations, which will eventually be replaced by fast rotations. Subsequently we incorporate the effects of the symmetric rank one formula (4) by the updates

$$C_{k+1} = C_{k+\frac{1}{2}} + \beta_k Y_{k+1} z_k z_k^T / z_k^T s_k \quad .$$

The total effort for bringing the five matrices Y_k, C_k, L_k, Z_k, and R_k up to date is clearly quadratic in $(n+m)$. For details see the fortcoming report [9]. A compact representation as discussed in [12] for unconstrained limited memory methods would be a possible alternative for very large problems.

6 Summary and Outlook

This paper probably raises more questions than it answers. However, our theoretical and experimental investigations do strongly suggest that by combining the direct and adjoint secant conditions (6) and (8) one obtains enough information to perform a rather promising two-sided-rank-one update (10), which we call TR1. The inclusion of adjoint information is not very expensive and makes this generalization of the well-known SR1 update to rectangular Jacobians completely invariant with respect to linear transformations on the domain and range of the underlying vector function, here the constraint $c(x)$. On affine problems the exact Jacobian is obtained in $m = \min(m, n)$

steps, unless the update formula breaks down prematurely due to a zero or very small denominator. Possible remedies for this contingency include damping factors and special steps in domain or dual range.

In the rather special situation $m = n$ the TR1 based coupled quasi-Newton iterations on the feasibility equation $c(x) = 0$ and a corresponding adjoint system $\nabla c(x)^T \lambda = -\nabla f(x)$ converge quite nicely. However, as indicated by the function values displayed in Figure 3 the convergence is not very steady, so that some damping of the update may be required. Newton's method is of course superior in terms of the iterations count but in general we would expect the exact Newton steps to be a lot more expensive to compute. The same is likely to be true for the full KKT system when $m < n$. On the very small set of equality constrained problems of Todd and Wagner we found that the total quasi Newton approach required maximally twice as many iterations as their mixed quasi-Newton approach. An implementation with quadratic complexity is under development and was sketched in Section 5.

Some important questions that immediately come to mind are the following.

1. In the affine square case $c(x) = A_* x - c_0$ and $f(x) = b^T x$, are there initial values of $A_0 \in \mathbb{R}^{n \times n}$ and $(x_0, \lambda_0) \in \mathbb{R}^{2n}$ for which our full quasi-Newton iteration reduces to an Arnoldi-like process?
2. How does one decide whether the SR1 and TR1 updates are too drastic and then dampen or completely skip them appropriately?
3. Are there update contenders for satisfying the three secant conditions $B_{k+1} s_k = w_k$, $A_{k+1} s_k = y_k$ and $A_{k+1}^T \sigma_k = \mu_k$ other than the seemingly natural SR1-TR1 combination advocated here?
4. How does one ensure convergence in nonlinear cases by coordinating the updates with line-searches and/or trust regions?
5. How does one deal with inequalities and determine active constraints?
6. How does a reasonable implementation of the advocated method fare against established codes on small to medium sized test problems?

With regards to the last question one must certainly expect and accept that the number of major iterations will be increased for a total quasi-Newton iteration. One may also be concerned that the lack of accurate Jacobian information will prolong the explicit or implicit procedure for determining the active constraints. Nevertheless, we are confident that a total quasi-Newton algorithm can be developed that is globally convergent under the usual assumption that $\text{rank}(\nabla c) = m$ and that is especially competitive when the Jacobian is rather expensive to form and factor explicitly.

7 Acknowledgment

The authors are very much indebted to the referees, who corrected numerous mistakes and suggested several improvements.

References

[1] R. Becker, R. Rannacher (2001). *An optimal control approach to a posteriori error estimation in finite element methods.* In Acta Numerica 2001 (A. Iserles, ed.), Cambridge University Press, 1–102.

[2] Å. Björck (1996). *An optimal control approach to error control and mesh adaptation in finite element methods.* SIAM, Philadelphia.

[3] J.H. Bramble and J.E. Pasciak (1988). A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Math. Comp*, 50:1–17.

[4] A.R. Conn and N.I.M. Gould and Ph.L. Toint (1991). Convergence of quasi-Newton matrices generated by the Symmetric Rank One update, *Mathematical Programming*, 50:177–196.

[5] J.E. Dennis, Jr., and R.B. Schnabel (1996). *Numerical methods for unconstrained optimization and nonlinear equations*, Classics Appl. Math., 16, SIAM, Philadelphia.

[6] A. Griewank (2001). Complexity of Gradients, Jacobians, and Hessians. *Encyclopedia of Optimization*. C.A. Floudas and P.M. Pardalos eds. Kluwer Academic Publishers, August.

[7] A. Griewank (2000). *Evaluating Derivatives, Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Appl. Math. SIAM, Philadelphia.

[8] A. Griewank (1987). The Local Convergence of Broyden's Method on Lipschitzian Problems in Hilbert Spaces. *SIAM Journal of Numerical Analysis*, 24:684–705.

[9] A. Griewank and A. Walther (2002). Maintaining factorized KKT Systems subject to Rank-one Updates of Hessians and Jacobians. In preparation.

[10] S.G. Nash and A. Sofer (1996). *Linear and nonlinear programming*, McGraw–Hill Series in Industrial Engineering and Management Science, McGraw–Hill, New York.

[11] U. Naumann (1999), *Efficient calculation of Jacobian matrices by optimized application of the chain rule to computational graphs*, Ph.D. thesis, Technical University Dresden.

[12] J. Nocedal and S.J. Wright (1999). Numerical Optimization, *Springer Series in Operation Research*. Springer Verlag, New York.

[13] M.J.D. Powell (1978). The convergence of variable metric methods for nonlinearly constrained optimization. In O.L. Mangasarin, R.R. Meyer and S.M. Robinson, editors, *Nonlinear Programming*, Academic Press, New York, pp 27–63.

[14] M.J.D. Powell (1970). A hybrid method for nonlinear equations. In *Numerical Methods for Nonlinear Algebraic Equations*, P. Rabinowitz, editor, Gordon & Breach, London, 77–114.

[15] M. Wagner and M.J. Todd (2000). *Least-change quasi-Newton updates for equality-constrained optimization*. Mathematical Programming. 87:317–350.

[16] W. Zulehner (2000). Analysis of Iterative Methods for Saddle Point Problems: a Uinfied Approach. AMS *Mathematics of Computation*. 71(238):479–505.