Computing sparse Hessian and Jacobian approximations with optimal hereditary properties

Roger Fletcher, Andreas Grothey and Sven Leyffer*

September 25, 1995

Abstract

In nonlinear optimization it is often important to estimate large sparse Hessian or Jacobian matrices, to be used for example in a trust region method. We propose an algorithm for computing a matrix $\mathbf{B}$ with a given sparsity pattern from a bundle of the m most recent difference vectors

$$\Delta = \left[\delta^{k-m+1} \dots \delta^k\right] \ , \ \Gamma = \left[\gamma^{k-m+1} \dots \gamma^k\right],$$

where $\mathbf{B}$ should approximately map Δ into Γ. In this paper $\mathbf{B}$ is chosen such that it satisfies m quasi–Newton conditions $\mathbf{B}\Delta = \Gamma$ in the least squares sense.

We show that $\mathbf{B}$ can always be computed by solving a positive semi–definite system of equations in the nonzero components of $\mathbf{B}$. We give necessary and sufficient conditions under which this system is positive definite and indicate how $\mathbf{B}$ can be computed efficiently using a conjugate gradient method.

In the case of unconstrained optimization we use the technique to determine a Hessian approximation which is used in a trust region method. Some numerical results are presented for a range of unconstrained test problems.

Keywords: Sparse nonlinear equations, sparse Hessian, limited memory, Procrustes Problems

1 Introduction

Consider the solution of large sparse optimization problems such as the minimization of a nonlinear function

$$\underset{x}{\text{minimize}}\, f(\mathbf{x}) \tag{1}$$

or the solution of a system of nonlinear equations

$$\mathbf{r}(\mathbf{x}) = \mathbf{0} \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ or $\mathbf{r} : \mathbb{R}^n \to \mathbb{R}^n$ are sufficiently smooth functions (i.e. $f \in \mathcal{C}^2$ and $\mathbf{r} \in \mathcal{C}^1$). Problems of this kind occur frequently in practice (e.g. Coleman [5]).

*University of Dundee, Department of Mathematics, Dundee, DD1 4HN, Scotland, U.K. (fletcher@mcs.dundee.ac.uk, sleyffer@mcs.dundee.ac.uk)

In many applications the gradient ∇f or the residual $\mathbf{r}$ are readily available, but the Hessian matrix $\nabla^2 f$ or the Jacobian matrix $\nabla \mathbf{r}^T$ cannot be conveniently computed. For instance, while the backward mode of Automatic Differentiation allows the gradient of a nonlinear function to be evaluated at a cost which is a small multiple of the cost of evaluating the function, the same does not hold for the Hessian (the cost of evaluating $\nabla^2 f$ is $\mathcal{O}(n)$ times the cost of evaluating f, e.g. [10]).

In the remainder of this section we concentrate on the case where a Hessian approximation is required. This is the more complex case. The question of approximating a sparse Jacobian is dealt with separately in Section 3.

The methods of choice for solving (1) are based on Newton's Method. However, since the Hessian matrix is not available it is not possible to apply Newton's method directly. If the Hessian involved is relatively small then quasi–Newton methods can be used to solve (1) (e.g.[6]). However, if the Hessian matrix is sparse, then these updates become prohibitive since the updated matrices fill–in, requiring the factorization of large dense matrices.

In this paper we aim to take advantage of the sparsity of the Hessian matrix. Let $\mathcal{S}$ denote the set of index pairs of the sparse or zero entries of $\nabla^2 f$. The sparsity conditions can then be expressed as

$$\left[\nabla^2 f\right]_{ij} = 0 \ , \ \text{if } (i,j) \in \mathcal{S}.$$

It is assumed that $\mathcal{S}$ is consistent with the symmetry of $\nabla^2 f$, i.e. $(i,j) \in \mathcal{S}$ if and only if $(j,i) \in \mathcal{S}$. We now seek an approximation $\mathbf{B} \simeq \nabla^2 f$ which satisfies the same sparsity condition.

For convenience, we also introduce the subset

$$\mathcal{T} := \{(i,j) \in \mathcal{S} \ , \ i \geq j\} \subset \mathcal{S}$$

of index pairs of the sparse entries of the *(lower) triangular part* of $\nabla^2 f$. The complement of $\mathcal{S}$, $\mathcal{S}^\perp$, is the set of index pairs of the non–sparse entries of $\nabla^2 f$. Similarly, $\mathcal{T}^\perp$ is the subset of $\mathcal{S}^\perp$ of lower triangular non–sparse entries.

In order to avoid handling the large dense Hessian matrices two alternatives to the classical quasi–Newton updates have been proposed. One employs a limited memory strategy storing only the m most recent difference vectors and performing the quasi–Newton update implicitly (Nocedal [12]) or a multiple quasi–Newton update (Barnes [4]). The multiple DFP, BFGS or SR1 updates also satisfy (3), but not the sparsity condition or symmetry. The other alternative takes advantage of the structure of large scale problems such as group partial separability (e.g. Conn, Gould and Toint [2]) or sparsity (e.g. Toint [15] or Fletcher [7]).

This paper is divided into 5 sections. The next section explains the key idea of the new Hessian approximation and shows how it can be computed. A sufficient condition for a unique approximation which does not require the diagonal to be non–sparse is developed and a related necessary condition is also investigated. Section 3 repeats the analysis of Section 2 for the case where a sparse Jacobian matrix is approximated. In Section 4 practical implementation issues are discussed and it is indicated how the Hessian approximation can be computed efficiently. Finally in Section 5 the results of a small number of numerical tests are presented and some conclusions are drawn.

2 Sparse Hessian approximations

In this section it is shown how a sparse Hessian approximation can be computed from the bundle of the m most recent difference pairs

$$\Delta = \left[\delta^{(k-m+1)} \dots \delta^{(k)}\right] \ , \ \Gamma = \left[\gamma^{(k-m+1)} \dots \gamma^{(k)}\right]$$

where $\delta^{(i)} = \mathbf{x}^{(i)} - \mathbf{x}^{(i-1)}$ and $\gamma^{(i)} = \nabla f^{(i)} - \nabla f^{(i-1)}$. The problem of estimating the entries of a sparse Hessian approximation leads in a natural way to a convex quadratic programming problem. First order optimality conditions for this problem are derived which lead to a symmetric and positive semi–definite system of linear equations in the coefficient of the sparse Hessian approximation. Finally, necessary and sufficient conditions for the linear system to be positive definite are investigated. A certain linear independence assumption is shown to be sufficient for positive definiteness. A related necessary condition is also investigated.

2.1 Key idea

The key idea is to seek a matrix $\mathbf{B}$ satisfying the same sparsity conditions as $\nabla^2 f$, i.e.

$$\mathbf{B}_{ij} = 0 \ , \ \text{if } (i,j) \in \mathcal{S}$$

which maps Δ into Γ, i.e.

$$\mathbf{B}\Delta = \Gamma. \tag{3}$$

In other words, $\mathbf{B}$ is required to satisfy m quasi–Newton conditions simultaneously.

If f is quadratic, then there always exists a matrix $\mathbf{B}$ that satisfies (3) and it is possible to reconstruct the exact Hessian from (3) provided a certain linear independence assumption holds. For non–quadratic functions however, one cannot expect to find a matrix $\mathbf{B}$ that solves (3) exactly for a given sparsity pattern. We therefore attempt to find a least squares solution to the matrix equation (3) by solving the following convex quadratic programming problem

$$CPP \left\{ \begin{array}{ll} \underset{B}{\text{minimize}} & \|\mathbf{B}\Delta - \Gamma\|_F^2 \\ \text{subject to} & \mathbf{B}^T = \mathbf{B} \\ & \mathbf{B}_{ij} = 0 \ , \ \forall (i,j) \in \mathcal{S} \end{array} \right.$$

where $\|X\|_F^2 = \text{trace}(XX^T) = \text{trace}(X^TX)$ denotes the squared Frobenius norm of X. This is often referred to as a Constrained Procrustes Problem (CPP) (e.g. Higham [9], see also Anderson and Elfving [1]). Since the constraints of (CPP) are linear and consistent and since the objective function is convex and bounded below by zero it follows that a solution to (CPP) always exists.

An interesting consequence of the simple structure of (CPP) is that it allows a straightforward generalization to the approximation of Hessian matrices which have some known elements. In the remainder of this section it is shown that a solution to (CPP) can be found by solving a sparse system of equations. Necessary and sufficient conditions under which the solution is unique are also discussed.

2.2 Solving the Procrustes problem

In this subsection it is shown how a solution to the Procrustes problem can be found by solving a sparse linear system in the unknown coefficients of $\mathbf{B}$. In order to solve (CPP) we introduce a matrix Λ of Lagrange multipliers for the symmetry constraints $\mathbf{B}^T = \mathbf{B}$ and a corresponding matrix Π for the sparsity constraints $\mathbf{B}_{ij} = 0 \ \forall (i,j) \in \mathcal{S}$. The Lagrangian of the Procrustes problem is then given by

$$\begin{aligned} \mathcal{L} &= \text{trace}\left((\mathbf{B}\Delta - \Gamma)(\mathbf{B}\Delta - \Gamma)^T\right) \\ &\quad + \text{trace}\left(\Lambda^T(\mathbf{B}^T - \mathbf{B})\right) \ + \ \text{trace}\left(\Pi^T\mathbf{B}\right) \end{aligned}$$

where the trace operator is used as a convenient way of summing over all constraints. Note that the complementarity condition implies that $\Pi_{ij} = 0$, if $(i,j) \in \mathcal{S}^{\perp}$.

Differentiating $\mathcal{L}$ with respect to $\mathbf{B}_{ij}$, the first order conditions provide the following equations

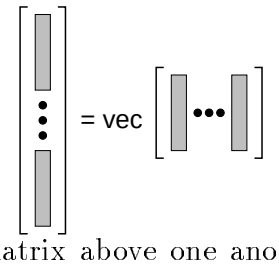
$$\begin{aligned} 0 = \frac{\partial \mathcal{L}}{\partial \mathbf{B}_{ij}} &= \text{trace}\left(\mathbf{B}\Delta\Delta^T e_j e_i^T\right) \ + \ \text{trace}\left(e_i e_j^T \Delta\Delta^T \mathbf{B}\right) \\ &\quad - \text{trace}\left(e_i e_j^T \Delta\Gamma^T\right) \ - \ \text{trace}\left(\Gamma\Delta^T e_j e_i^T\right) \\ &\quad + \text{trace}\left(\Lambda^T(e_i e_j^T - e_j e_i^T)\right) \ + \ \text{trace}\left(\Pi^T e_i e_j^T\right) \\ &= \left[\mathbf{B}\Delta\Delta^T\right]_{ij} + \left[\Delta\Delta^T\mathbf{B}\right]_{ij} - \left[\Delta\Gamma^T\right]_{ij} - \left[\Gamma\Delta^T\right]_{ij} + [\Lambda]_{ij} - [\Lambda]_{ji} + [\Pi]_{ij} \end{aligned}$$

by virtue of the invariance of the trace of a product to cyclic permutations and transposition. Adding the transpose of this last equation to itself and using the symmetry of $\mathbf{B}$ gives rise to the matrix equation

$$\mathbf{B}\Delta\Delta^T + \Delta\Delta^T\mathbf{B} - \Delta\Gamma^T - \Gamma\Delta^T + \Pi = 0. \tag{4}$$

This equation is sometimes referred to as the Lyapunov equation (see Ortega [13, p. 248]). Note that equation (4) may be underdetermined, since $\Delta\Delta^T$ is a low rank matrix. Since (CPP) is a convex quadratic programming problem, feasibility and the first order condition imply the existence of a solution of (4). Necessary and sufficient conditions under which (4) has a *unique* solution depend on the sparsity structure of $\nabla^2 f$ as well as on certain properties of Δ and these conditions are presented in the next subsection.

In order to gain more insight into (4) it is convenient to vectorize the matrix equation. Therefore the vec operator is introduced



which stacks the columns of a matrix above one another. The inverse mapping to vec which transforms an n^2 vector into an $n \times n$ matrix is denoted by vec^{-1}. Defining vector versions of the matrices involved in (4)

$$\mathbf{b} = \text{vec}(\mathbf{B}) \ , \ \boldsymbol{\pi} = \text{vec}(\Pi) \ , \ \mathbf{w} = \text{vec}(\Delta\Gamma^T + \Gamma\Delta^T)$$

it is possible to write (4) as the $n^2 \times n^2$ system

$$\mathbf{K}\mathbf{b} + \boldsymbol{\pi} = \mathbf{w} \tag{5}$$

where $\mathbf{K} = (\Delta\Delta^T \otimes \mathbf{I}) + (\mathbf{I} \otimes \Delta\Delta^T)$ is the Kronecker sum of $\Delta\Delta^T$ with itself. The first matrix $\Delta\Delta^T \otimes \mathbf{I}$ is block diagonal with n blocks $\Delta\Delta^T$ whilst the second matrix is a permutation of the first, having n^2 blocks each of which is a multiple of the identity matrix (the ij–th block is $[\Delta\Delta^T]_{ij}\mathbf{I}$ (see e.g. Lancaster and Tismenetzky [11])).

Equation (5) includes the computation of the known zero elements of $\mathbf{B}$ and of the Lagrange multipliers Π which are not needed in the Hessian approximation. The multipliers and the sparse elements are therefore removed from (5), resulting in a sparse linear system in the non–sparse elements of $\mathbf{B}$ only.

It is also useful to introduce

$$\Omega := \left\{ \mathbf{v} : \ \mathbf{v} = \text{vec}(\mathbf{V}),\ \mathbf{V}^T = \mathbf{V},\ \mathbf{V}_{ij} = 0 \ \forall (i,j) \in \mathcal{S} \right\}$$

which is the linear space of symmetric and sparse vectors. We now seek a solution of (5) which lies in Ω. For example consider the sparsity pattern of $\nabla^2 f$ given in Figure 1. Although $\mathbf{K}$ is a 9×9 matrix only the five $\bullet$ elements of $\mathbf{b}$ have to be computed. The $\bullet$ elements of π act as slacks for the equations defining the corresponding zero elements in $\mathbf{b}$.

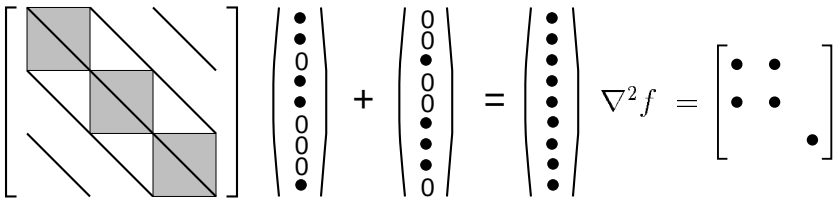


Figure 1: Example of Kronecker System

In fact, using the symmetry of $\nabla^2 f$ a further reduction is possible to eliminate the entries of $\mathbf{B}$ which are strictly upper triangular (i.e. $\mathbf{B}_{ij}$ for $i < j$). Seeking a solution of (5) in Ω is equivalent to projecting (5) onto the non–sparse lower triangular elements of $\mathbf{b}$. This reduction of the problem can be expressed by introducing the matrix $\mathbf{P}$ whose columns are made up of the vectors

$$\text{vec}(E_{ij} + E_{ji}) \ , \ \forall (i,j) \in \mathcal{T}^{\perp} \quad \text{and} \quad \text{vec}(E_{ii}) \ , \ \forall (i,i) \in \mathcal{T}^{\perp}$$

where E_{ij} is the matrix with 1 in (i,j) position and zeros elsewhere. Introducing vectors $\mathbf{b}'$ and $\mathbf{w}'$ corresponding to the non–sparse lower triangular entries of $\mathbf{b}$, equation (5) is equivalent to

$$\mathbf{P}^T\mathbf{K}\mathbf{P}\mathbf{b}' = \mathbf{w}' \tag{6}$$

Since $\Delta\Delta^T$ is positive semi–definite it follows from [11, Corollary 12.2.2] that the system (6) is positive semi–definite. It is of interest to know, under what circumstances $\mathbf{P}^T\mathbf{K}\mathbf{P}$ is positive definite, or equivalently, when a unique Hessian approximation $\mathbf{B}$ can be computed from (6). This question is also equivalent to asking under what conditions $\mathbf{K}$ is positive definite on Ω (i.e. $\mathbf{v}^T\mathbf{K}\mathbf{v} > 0$, $\forall \mathbf{v} \in \Omega$, $\mathbf{v} \neq \mathbf{0}$).

Example: Tridiagonal Hessian
To illustrate the development so far the estimation of a tridiagonal Hessian matrix

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdot \\ b_{21} & b_{22} & b_{23} \\ \cdot & b_{32} & b_{33} \end{bmatrix} \text{ from } \Delta = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 1 & 1 \end{bmatrix}, \Gamma = \begin{bmatrix} 4 & 1 \\ 2 & 0 \\ 1 & 4 \end{bmatrix}$$

is considered. Forming the Kronecker products and using the symmetry of $\mathbf{B}$ and Π, the following Kronecker system (5) is obtained.

$$\left[\begin{array}{ccc|ccc|ccc} 2 & 2 & 1 & 2 & \cdot & \cdot & 1 & \cdot & \cdot \\ 2 & 6 & 3 & \cdot & 2 & \cdot & \cdot & 1 & \cdot \\ 1 & 3 & 3 & \cdot & \cdot & 2 & \cdot & \cdot & 1 \\ \hline 2 & \cdot & \cdot & 6 & 2 & 1 & 3 & \cdot & \cdot \\ \cdot & 2 & \cdot & 2 & 10 & 3 & \cdot & 3 & \cdot \\ \cdot & \cdot & 2 & 1 & 3 & 7 & \cdot & \cdot & 3 \\ \hline 1 & \cdot & \cdot & 3 & \cdot & \cdot & 3 & 2 & 1 \\ \cdot & 1 & \cdot & \cdot & 3 & \cdot & 2 & 7 & 3 \\ \cdot & \cdot & 1 & \cdot & \cdot & 3 & 1 & 3 & 4 \end{array}\right] \left(\begin{array}{c} b_{11} \\ b_{21} \\ \cdot \\ b_{21} \\ b_{22} \\ b_{32} \\ \cdot \\ b_{32} \\ b_{33} \end{array}\right) + \left(\begin{array}{c} \cdot \\ \cdot \\ \pi_{31} \\ \cdot \\ \cdot \\ \cdot \\ \pi_{31} \\ \cdot \\ \cdot \end{array}\right) = \left(\begin{array}{c} 8 \\ 11 \\ 6 \\ 11 \\ 8 \\ 8 \\ 6 \\ 8 \\ 1 \end{array}\right)$$

The corresponding reduced system (6) is given by

$$\begin{bmatrix} 2 & 4 & \cdot & \cdot & \cdot \\ 4 & 12 & 4 & 2 & \cdot \\ \cdot & 4 & 10 & 6 & \cdot \\ \cdot & 2 & 6 & 14 & 6 \\ \cdot & \cdot & \cdot & 6 & 4 \end{bmatrix} \begin{pmatrix} b_{11} \\ b_{21} \\ b_{22} \\ b_{32} \\ b_{33} \end{pmatrix} = \begin{pmatrix} 8 \\ 22 \\ 8 \\ 16 \\ 10 \end{pmatrix}$$

which can be seen to be positive definite. Thus $\mathbf{K}$ is positive definite on Ω and a unique Hessian approximation can be found. It is readily shown however, that for

$$\Delta = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$\mathbf{K}$ is *not* positive definite on Ω. Thus linear independence of the columns of Δ does not on its own ensure positive definiteness of $\mathbf{K}$ on Ω. A stronger condition is required and this is discussed in the next subsection.

2.3 Conditions for a unique B

It is pointed out in the previous section that the linear system (6) is positive semi–definite. Here necessary and sufficient conditions which ensure that $\mathbf{K}$ is positive–definite on Ω are discussed.

Let

$$\mathcal{T}_j^{\perp} := \left\{ i : (i,j) \in \mathcal{T}^{\perp} \right\}$$

be the index set of the non–zero elements in column j of the lower triangular part of $\nabla^2 f$. Moreover denote the columns and rows of Δ by

$$\Delta = \left[\delta^{(k-m+1)} \dots \delta^{(k)} \right] = \begin{bmatrix} \Delta_1^T \\ \vdots \\ \Delta_n^T \end{bmatrix}.$$

A sufficient condition that $\mathbf{K}$ is positive–definite on Ω is then given by the following theorem.

Theorem 1 *If there exists an ordering of the variables* $\mathbf{x}$ *such that for each* $j = 1, 2, \ldots, n$, *the row vectors* $\Delta_i^T \ : \ i \in \mathcal{T}_j^{\perp}$ *are linearly independent, then* $\mathbf{K}$ *is positive definite on* Ω.

Proof:
Proof by contradiction. Let the conditions of Theorem 1 be satisfied and assume that $\mathbf{K}$ is *not* positive definite on Ω. Then there exists a $\mathbf{v} \in \Omega$, $\mathbf{v} \neq \mathbf{0}$ such that $\mathbf{v}^T\mathbf{K}\mathbf{v} = 0$.

Since $\mathbf{K}$ is the sum of two positive semi–definite matrices, it follows that $\mathbf{v}^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{v} = 0$. Likewise since $(\mathbf{I} \otimes \Delta\Delta^T) = \mathrm{diag}(\Delta\Delta^T)$ is block diagonal and Δ has full column rank it follows that

$$\Delta^T \mathbf{v}_1 = \mathbf{0}$$

where $\mathbf{v}_1$ is the first column of $\mathbf{V} = \mathrm{vec}^{-1}(\mathbf{v})$. This last equation implies that

$$\sum_{i \in \mathcal{T}_j^{\perp}} \Delta_i v_{i1} = \mathbf{0}.$$

Since the vectors $\Delta_i^T \ : \ i \in \mathcal{T}_j^{\perp}$ are linearly independent, it follows that $v_{i1} = 0 \ \forall i$. Since $\mathbf{V}$ is symmetric it follows that $v_{12} = v_{21} = 0$. Repeating the above argument inductively for columns $2, \ldots, n$ of $\mathbf{V}$ it follows that $\mathbf{v} = \mathbf{0}$ which contradicts the assumption. It therefore follows that $\mathbf{K}$ is positive definite on Ω. □

Note that unlike the results in Toint [15] or Fletcher [7], Theorem 1 does not assume that the diagonal of $\nabla^2 f$ is non–sparse, i.e. we do not need to assume that $(i, i) \in \mathcal{S}^{\perp}$.

Example 1 revisited:
The sets $\mathcal{T}_j^{\perp}$ are $\{1, 2\}$, $\{2, 3\}$ and $\{3\}$. Since the corresponding rows of Δ are linearly independent it follows from Theorem 1 that $\mathbf{K}$ is positive definite on Ω.

Theorem 1 also shows that 2 difference pairs are sufficient to estimate an arrowhead matrix since it can be reordered so that there are at most two entries in each column of the lower triangular part. From Figure 2 it can be seen that a sufficient condition for positive definiteness is that rows of Δ corresponding to the sets $\{1, n\}$, $\{2, n\}$, ... , $\{n-1, n\}$ are linearly independent.

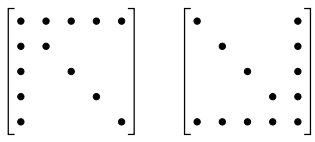


Figure 2: Arrowhead matrix and re–ordering

The next theorem shows that the sufficient condition of Theorem 1 is also almost a necessary condition.

Theorem 2 *Assume that* $(i, i) \in \mathcal{S}^{\perp}$, $\forall i$. *If there exists an ordering of the variables* $\mathbf{x}$ *such that no fill–in occurs when forming* LDL^T *factors of* $\nabla^2 f$ *and if* $\mathbf{K}$ *is positive definite on* Ω, *then for each* $j = 1, 2, \ldots, n$ *the vectors* $\Delta_i^T \ : \ i \in \mathcal{T}_j^{\perp}$ *are linearly independent.*

Proof:
Proof by contradiction. Assume that there exists an index j such that the vectors $\Delta_i^T \ : \ i \in \mathcal{T}_j^\perp$ are linearly dependent. Let j_0 be the largest such index. Then there exists a vector $\mathbf{v}' \neq \mathbf{0}$ such that

$$\sum_{i \in \mathcal{T}_{j_0}^\perp} \Delta_i v_i' = \mathbf{0}.$$

Now we show that $\mathbf{v}'$ may be completed to an n^2 non–zero vector $\mathbf{v}$ for which $\mathbf{v}^T\mathbf{K}\mathbf{v} = 0$, giving the desired contradiction.

Set the first $j_0 - 1$ columns of $\mathbf{v}$ to zero ($\mathbf{v}_1 = \mathbf{v}_2 = \ldots = \mathbf{v}_{j_0-1} = \mathbf{0}$) and let column j_0 be defined by

$$v_{i,j_0} = \begin{cases} 0 & \text{for } i \in \mathcal{T}_{j_0} \\ v_i' & \text{for } i \in \mathcal{T}_{j_0}^\perp \end{cases}$$

Then $\mathbf{v}_{j_0} \neq \mathbf{0}$ and by construction $\Delta^T\mathbf{v}_{j_0} = \mathbf{0}$. The remaining entries of $\mathbf{v}$ are now completed according to the sparsity pattern of $\nabla^2 f$ as follows

$$v_{i,j} = \frac{v_{i,j_0} v_{j_0,j}}{v_{j_0,j_0}} \ \forall i, j > j_0.$$

The assumption that no fill–in occurs when forming LDL^T factors of $\nabla^2 f$ ensures that the index pairs (i, j) corresponding to these $v_{i,j}$ are in $\mathcal{S}^\perp$ so that the resulting $\mathbf{V}$ is in Ω. This fill-in corresponds to the trailing $(n - j_0) \times (n - j_0)$ Markowitz submatrix of [7]. Note that $v_{j_0,j_0} \neq 0$ since j_0 is the *largest* index such that the vectors $\Delta_i^T \ : \ i \in \mathcal{T}_j^\perp$ are linearly dependent (if v_{j_0,j_0} were zero, there would be a larger index for which linear dependence would hold). Clearly, $\mathbf{v} = \text{vec}(\mathbf{V}) \neq \mathbf{0}$ and $\mathbf{v} \in \Omega$, since it has the same sparsity pattern as $\nabla^2 f$ and is symmetric. Now it follows that

$$\mathbf{v}^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{v} = \sum_{j=1}^{n} \mathbf{v}_j^T \Delta\Delta^T \mathbf{v}_j$$

The first $j_0 - 1$ terms in this sum are zero since $\mathbf{v}_1 = \ldots = \mathbf{v}_{j_0-1} = \mathbf{0}$. Moreover, by construction $\Delta^T\mathbf{v}_{j_0} = \mathbf{0}$. The remaining $n - j_0$ terms are also zero, since

$$\Delta^T\mathbf{v}_i = \frac{v_{i,j_0}}{v_{j_0,j_0}} \Delta^T\mathbf{v}_{j_0} = \mathbf{0} \ \forall i > j_0$$

Thus it follows that

$$\mathbf{v}^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{v} = 0.$$

Since $\mathbf{v}$ is symmetric and since $(\Delta\Delta^T \otimes \mathbf{I})$ is a symmetric permutation of $(\mathbf{I} \otimes \Delta\Delta^T)$, it follows that also

$$\mathbf{v}^T(\Delta\Delta^T \otimes \mathbf{I})\mathbf{v} = 0.$$

Thus $\mathbf{v}^T\mathbf{K}\mathbf{v} = 0$ which contradicts the assumption that $\mathbf{K}$ is positive definite on Ω and therefore it follows that the vectors $\Delta_i^T \ : \ i \in \mathcal{T}_i^\perp$ are linearly independent $\forall j$. □

Theorem 2 gives the minimum number of difference pairs required to estimate a Hessian matrix with a given sparsity pattern, namely the minimum of

$$\max_j |\mathcal{T}_j^\perp|$$

taken over all possible reorderings such that no fill–in occurs. In the case of Example 1 this means that at least 2 difference pairs are required to estimate the tridiagonal Hessian (the same number as for an arrowhead matrix). Moreover, since for

$$\Delta = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

the rows corresponding to $\mathcal{T}_2^{\perp} = \{2, 3\}$ are linearly dependent, it follows from Theorem 2 that $\mathbf{K}$ is not positive definite on Ω.

Theorem 2 shows that if the vectors $\Delta_i : \ i \in \mathcal{T}_j^{\perp}$ are linearly dependent then not all elements of row/column j of $\mathbf{B}$ are well determined. However, in this case a subset of column entries can be computed, provided that the corresponding subset of Δ_i's is linearly independent. In this sense our result differs from Toint's [15]. In Toint's case a zero Δ_i implied that *no* entry in the corresponding row/column of the Hessian could be computed whereas our procedure still allows other (off–diagonal) elements of the Hessian to be estimated.

2.4 Quadratic termination

If the objective function f is a quadratic and the matrix Δ is such that for each $j = 1, \ldots, n$ the row vectors $\Delta_i^T \ : \ i \in \mathcal{T}_j^{\perp}$ are linearly independent, then it follows that the matrix $\mathbf{B}$ that solves (CPP) is the exact Hessian itself. In this case it is possible for an optimization routine based on the new Hessian approximation to terminate in at most $\max_j |\mathcal{T}_j^{\perp}|$ steps.

Using the $\| \cdot \|_2$ norm by Toint [15] gives the sparse analogue of the PSB update and does not usually have quadratic termination. The use of a weighted norm (e.g. [7]) gives much added complication in the sparse case. The approach in our paper is notable in that it allows for quadratic termination whilst retaining the simplicity of the $\| \cdot \|_2$ norm.

3 Sparse Jacobian approximations

In this section the results of the previous section are generalized to the case where a sparse (non–symmetric) Jacobian matrix is approximated. The set of index pairs $\mathcal{S}$ is now no longer symmetric. In this case the following Procrustes problem is considered

$$\begin{cases} \underset{B}{\text{minimize}} & \|\mathbf{B}\Delta - \Gamma\|_F^2 \\ & \mathbf{B}_{ij} = 0 \ , \ \forall (i,j) \in \mathcal{S} \end{cases}$$

where $\mathcal{S}$ now denotes the sparsity pattern of $\nabla \mathbf{r}^T$. Note that the columns $\gamma^{(i)}$, $i = k - m + 1, \ldots, k$ of Γ correspond to differences $\mathbf{r}^{(i)} - \mathbf{r}^{(i-1)}$ in the residual vector.

In a similar way to the previous section it is possible to derive first order conditions for this Procrustes problem by introducing a Lagrange multiplier matrix Π for the sparsity constraints. Differentiating the Lagrangian leads to a matrix equation similar to (4), namely

$$\mathbf{B}\Delta\Delta^T \ - \ \Gamma\Delta^T \ + \ \Pi = 0 \tag{7}$$

By vectorizing equation (7) the following symmetric and positive semi–definite linear system is obtained

$$\widehat{\mathbf{K}}\widehat{\mathbf{b}} \ + \ \widehat{\boldsymbol{\pi}} \ = \ \widehat{\mathbf{w}} \tag{8}$$

where $\widehat{\mathbf{K}} = (\Delta\Delta^T \otimes \mathbf{I})$ and $\hat{\mathbf{w}} = \mathrm{vec}(\Gamma\Delta^T)$. Only the non–zero elements of $\mathbf{b}$ are of interest and therefore (8) is projected onto the non–sparse elements. Equivalently, a solution of (8) is sought which lies in the set of sparse vectors

$$\widehat{\Omega} := \{\mathbf{v} : \ \mathbf{v} \ = \ \mathrm{vec}(\mathbf{V}), \ \mathbf{V}_{ij} = 0 \ \forall (i,j) \in \mathcal{S}\}$$

Defining the index set of non–zeros of each row of $\nabla\mathbf{r}^T$

$$\mathcal{S}_i^{\perp} \ := \ \left\{j \ : \ (i,j) \in \mathcal{S}^{\perp}\right\}$$

the following characterization of a unique solution to (8) can be obtained.

Theorem 3 *$\widehat{\mathbf{K}}$ is positive definite on $\widehat{\Omega}$ if and only if the vectors $\Delta_j^T \ : \ j \in \mathcal{S}_i^{\perp}$ are linearly independent for all $i = 1, 2, \ldots, n$.*

Proof:
$\widehat{\mathbf{K}} = (\mathbf{I} \otimes \Delta\Delta^T) = \mathrm{diag}(\Delta\Delta^T, \ldots, \Delta\Delta^T)$ is clearly positive semi–definite.

Now define matrices $\mathbf{P}_i$ whose columns are the unit vectors $\mathbf{e}_j \in \mathbb{R}^n$ for $j \in \mathcal{S}_i^{\perp}$ and assume that $\mathcal{S}_i^{\perp} = \{i_1, \ldots, i_k\}$. Then it follows that $\mathbf{P} := \mathrm{diag}(\mathbf{P}_1, \ldots, \mathbf{P}_n)$ reduces (8) to a system in the non–sparse elements. Moreover

$$\mathbf{P}^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{P} = \mathrm{diag}(\mathbf{P}_1^T\Delta\Delta^T\mathbf{P}_1^T, \ldots, \mathbf{P}_n^T\Delta\Delta^T\mathbf{P}_n^T)$$

and the following equivalences can be established.

$$\begin{aligned}
& \mathbf{P}^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{P} \text{ is positive definite} \\
\Leftrightarrow \quad & \mathbf{P}_i^T(\mathbf{I} \otimes \Delta\Delta^T)\mathbf{P}_i \text{ is positive definite } \forall i = 1, \ldots, n \\
\Leftrightarrow \quad & \begin{bmatrix} \Delta_{i_1} \\ \vdots \\ \Delta_{i_k} \end{bmatrix} [\Delta_{i_1} \ldots \Delta_{i_k}] \text{ is positive definite } \forall i = 1, \ldots, n \\
\Leftrightarrow \quad & \langle \Delta_j^T \ : \ j \in \mathcal{S}_i^{\perp} \rangle \text{ are linearly independent } \forall i = 1, \ldots, n
\end{aligned}$$

□

Note that Theorem 3 shows that a necessary condition for a unique Jacobian approximation is that the number of difference pairs satisfies $m \ \geq \ \max_i |\mathcal{S}_i^{\perp}|$.

Curtis, Powell and Reid [3] propose a similar scheme for estimating the Jacobian, which was extended by Powell and Toint [14] to the case of a symmetric Hessian. However, there are a number of important differences which are pointed out now. Both [3] and [14] prescribe the differencing directions $\delta^{(i)}$ and require one extra gradient/residual evaluation per $\delta^{(i)}$ each time the Hessian/Jacobian is approximated. Moreover, unless substitution methods are used their schemes require more difference pairs than our scheme. The new scheme, on the other hand, takes difference vectors generated by the optimization algorithm. As a consequence of solving (CPP), the new scheme has the property that it generates the Hessian/Jacobian approximation which best fits the m quasi–Newton conditions inherent in the difference pairs.

4 Practical implementation

In this section practical implementation issues of the Hessian approximation scheme are discussed. Most comments carry over directly to the case of a Jacobian approximation scheme.

The new Hessian approximation is not positive definite in general. It is therefore used in a trust–region algorithm. At each iteration of the trust–region algorithm a new Hessian approximation is computed.

A consequence of the linear independence assumption of Theorem 1 is that the number of elements per column of the lower triangular part of $\nabla^2 f$ that can be estimated is limited by m. In our implementation at the first step, we therefore estimate only the n diagonal entries of $\mathbf{B}$. After the second iteration the diagonal and one off–diagonal entry per column can be estimated. Proceeding in this fashion a bundle of difference pairs is build up, until sufficient difference pairs are accumulated in Δ and Γ. At this point the oldest difference pair is replaced by a new one, similar to a limited memory strategy.

In the remainder of this section a practical way of solving the linear system (6) is discussed. This step is non–trivial for the following two reasons. Firstly, the matrix $\mathbf{P}^T\mathbf{K}\mathbf{P}$ is very large; it's size is $L \times L$ where L is the number of non–zeros in the lower triangular part of $\nabla^2 f$. A dense factorization of $\mathbf{P}^T\mathbf{K}\mathbf{P}$ would therefore be too expensive ($\mathcal{O}(L^3)$ flops, where $L \geq n$ usually) to contemplate. On the other hand, unlike in [15] or [7] $\mathbf{P}^T\mathbf{K}\mathbf{P}$ does not seem to have any obvious sparsity structure of which one could take advantage. We are currently investigating alternative direct methods for solving (6). Secondly, the positive definiteness condition of Theorem 1 is cumbersome to verify in practice. Thus there appears to be no easy and stable way of detecting singularity in the positive semi–definite matrix. Higham shows that the Cholesky algorithm for semi–definite matrices with complete pivoting is stable [8] so that there is some hope that the reduced system can be solved accurately and efficiently.

While the two above points argue against solving (6) by a direct method it is possible to form efficient matrix vectors products with $\mathbf{K}$. In the current implementation we therefore solve (6) by conjugate gradients. Each iteration of the trust region algorithm requires the solution of a (CPP) problem. For this problem, the conjugate gradient iteration is initialized with the Hessian approximation from the previous (CPP) problem. It is hoped that this provides a good starting point since only one column of Δ changes. It is an open question whether it is possible to take more direct advantage of the change in Δ to find subsequent Hessian approximations.

The use of an iterative solver for (6) has the additional advantage that it is possible to solve (6) inexactly. This can result in savings in CPU time early on during the minimization where an accurate Hessian approximation is not required. As the trust region algorithm converges (6) is then solved to a higher accuracy. We have experimented with one possible such scheme and the results are reported in the next section.

4.1 Conjugate gradients to solve (6)

In order to solve (6) by conjugate gradients, efficient evaluation of the product $\mathbf{P}^T\mathbf{K}\mathbf{P}\,\mathbf{v}$ is frequently required. In this section it is shown how this product can be computed efficiently *without* forming $\mathbf{P}^T\mathbf{K}\mathbf{P}$ explicitly.

Efficient products with $\mathbf{P}^T\mathbf{K}\mathbf{P}$ can be obtained by "unravelling" the definitions of

$\mathbf{P}$ and $\mathbf{K}$. Clearly, it is equivalent to form $\mathbf{K}\,\mathbf{v}$ for the non–sparse elements only. This in turn is equivalent to computing $\Delta\Delta^T\mathbf{V} + \mathbf{V}\Delta\Delta^T$ for a sparse and symmetric matrix $\mathbf{V} = \mathrm{vec}^{-1}(\mathbf{v})$ in Ω which has the same sparsity pattern as $\nabla^2 f$. In practice it is sufficient to operate with Δ^T on $\mathbf{V}$, forming $\Delta\Delta^T\mathbf{V}$ and to add the transpose.

The number of floating point operations for one such product is less than $7mL$ where m is the number of difference pairs and L is the number of non–zeros in the lower triangular part of $\nabla^2 f$. The corresponding number of flops for the Jacobian approximation is $2mL$, where L is the number of *all* non–zeros of $\nabla \mathbf{r}^T$.

5 Numerical experience and conclusions

An experimental trust region code which uses the new Hessian approximation for minimizing a nonlinear function has been implemented in Matlab. We have tested the method on two problems; a chained Rosenbrock problem and a boundary value problem [7] for $n = 10$ and $n = 100$ variables. In both cases, the tridiagonal Hessian is approximated using the $m = 2$ most recent difference pairs. The conjugate gradient iterations are terminated when the residual of the conjugate gradient solve is less than $\|\nabla f^i\|$, where i is the current trust region iteration. The new method is compared to Nocedal's low storage method [12] (storing $m = 2$ and $m = 5$ difference pairs) with a line–search and to Newton's method with a trust region. We have implemented an inexact line–search for Nocedal's method, since experiments with an exact line–search showed that it saves only a few iterations while requireing about twice as many function/gradient vealuations as an inexact line–search.

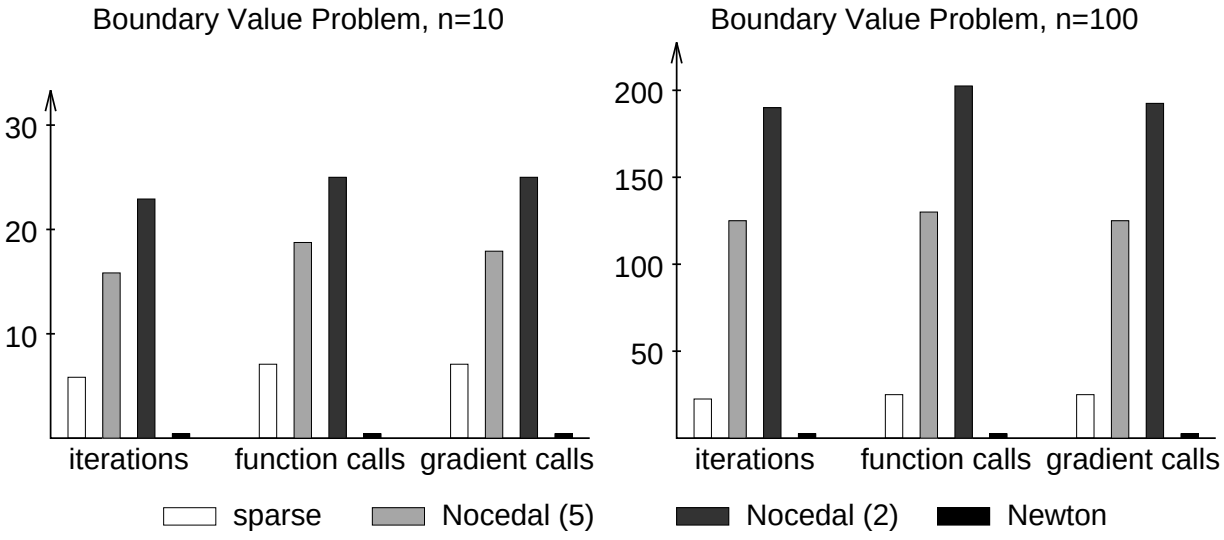


Figure 3: Results for boundary value problem

The results for the two test are displayed in Figures 3 and 4 (sparse refers to the trust region method with the new Hessian approximation). The results show that the new method can be made to work and that it is competitive with Nocedal's method (especially, if compared to Nocedal (2) which has the same storage requirements). Overall, it is disappointing that the new method does not show a marked improvement over Nocedal's

method (as we would have expected, since the new method utilizes more information (sparsity structure) than Nocedal's method). The results for Newton's method represent a kind of performance goal for any other Newton–like method. It should not come as a surprise that Newton's method is the best of the three methods since it utilizes the most problem information (exact Hessian) of the three methods.

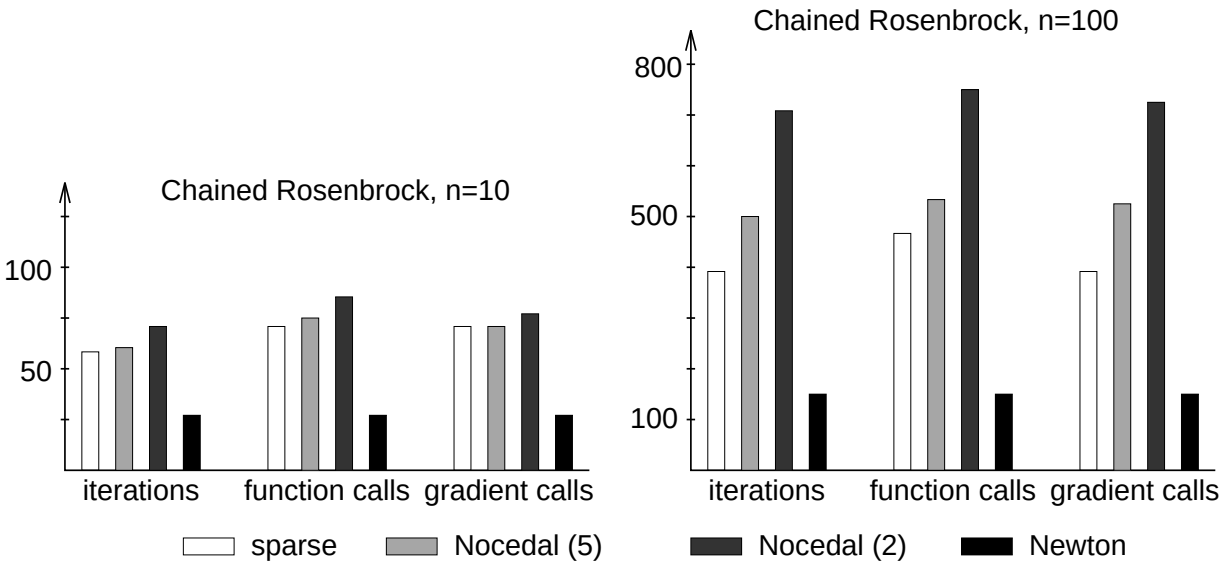


Figure 4: Results for chained Rosenbrock's problem

In the experiment, the new method required between 5 and 10 conjugate gradient iterations per trust–region iteration to compute the new Hessian approximation. Since each conjugate gradient iteration costs about $7mL$ flops, the total number of flops to compute the Hessian is about $70mL$. This is an order of magnitude larger than the cost per iteration for Nocedal's method (namely $4mn$, where usually $n < L$). The function and gradient calls are slightly higher for Nocedal's method due to the line–search which requires more function/gradient calls than a trust–region method per iteration. On the other hand, one iteration of the trust–region algorithm is usually more expensive than one line–search, so that a comparison based solely on the number of iterations can only give a rough idea of the usefulness of the new method.

Overall, we have been rather disappointed by these preliminary experiments and one direction of further research will focus on the question as to how the method can be made to work more efficiently. Another future direction of research is the implementation of the Hessian approximation in an SQP method for nonlinear programming problems, where a trust–region method would be our method of choice. The new Hessian approximation appears to be well suited to this situation since for example we do not assume that the diagonal entries of the Hessian are non–sparse. Moreover, the simple data structure for $\mathbf{K}$ makes it easy to accommodate changes in the active set.

References

[1] L.-E. Anderson and T. Elfving, *A Constrained Procrustes Poblem*, Technical

Report LiTH-MAT-R-1993-39, Department of Mathematics, Linköping University, S-581 83 Linköpin, Sweden, November 1994. Submitted to SIAM Journal on Matrix Analysis and Applications.

[2] A.R. Conn, N.I.M. Gould and Ph.L. Toint, *A proposal for a standard data input format for large–scale nonlinear programming problems*, Technical Report CS-89-61, University of Waterloo, Computer Science Department, Ontario, Canada, March 1990.

[3] A.R. Curtis, M.J.D. Powell and J.K. Reid, *On the estimation of sparse Jacobian matrices*, *Journal of the Institute of Mathematics and its Applications*, 13:117–120, 1974.

[4] J. Barnes, *An algorithm for solving nonlinear equations based on the secand method*, *Computer Journal*, 8:66–67, 1965.

[5] Thomas F. Coleman, *Large-scale numerical Optimization: Introduction and Overview*, Technical Report CTC91TR85, Cornell Theory Center, Cornell University, Ithaca, NY 14853–5201, September 1991.

[6] R. Fletcher, *Practical Methods of Optimization, 2^{nd} edition*, John Wiley, Chichester, 1987.

[7] R. Fletcher, *An optimal positive definite update for sparse hessian matrices*, *SIAM Journal on Optimization*, 5(1):192–218, February 1995.

[8] N. J. Higham, *Analysis of the Cholesky decomposition of a semi–definite matrix*, In M.G. Cox and S. Hammarling, editors, *Reliable Numerical Computation*, pages 161–185, Oxford, 1990. Oxford University Press.

[9] N.J. Higham, *The Symmetric Procrustes Problem*, *BIT*, 28:133–143, 1988.

[10] M. Iri, *History of automatic differentiation and rounding error estimation*, In A. Griewank and G.F. Corliss, editors, *Automatic Differentiation of Algorithms*, pages 3–24, Philadelphia, 1991. SIAM.

[11] P. Lancaster and M. Tismenetsky, *The Theory of Matrices, Second Edition*, Computer Science and Applied Mathematics. Academic Press, New York, 1985.

[12] J. Nocedal, *Updating quasi–Newton matrices with limited storage*, *Mathematics of Computation*, 35:773–782, 1980.

[13] J.M. Ortega, *Matrix Theory*, Plenum Press, New York, 1987.

[14] M.J.D Powell and Ph.L. Toint, *On the estimation of sparse Hessian matrices*, *SIAM Journal on Numerical Analysis*, 16(6):1060–1074, December 1979.

[15] Ph.L. Toint, *On sparse and symmetric matrix updating subject to a linear equation*, *Mathematics of Computation*, 31(140):954–961, October 1977.