**REPORTS
IN
INFORMATICS**

ISSN 0333-3590

Sparsity Issues in the Computation of Jacobian Matrices

Shahadat Hossain and Trond Steihaug

REPORT NO 223 **January 2002**

Department of Informatics
UNIVERSITY OF BERGEN
Bergen, Norway

This report has URL `http://www.ii.uib.no/publikasjoner/texrap/ps/2002-223.ps`

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at `http://www.ii.uib.no/publikasjoner/texrap/`.

Requests for paper copies of this report can be sent to:

Department of Informatics, University of Bergen, Høyteknologisenteret, P.O. Box 7800, N-5020 Bergen, Norway

Sparsity Issues in the Computation of Jacobian Matrices

Shahadat Hossain *

Department of Mathematics and Computer Science
University of Lethbridge
AB T1K 4G9 Canada
and
Trond Steihaug†

Department of Informatics
University of Bergen
N-5020 Bergen
Norway

15th January 2002

Abstract

The knowledge of sparsity information plays an important role in efficient determination of sparse Jacobian matrices. In a recent work, we have proposed sparsity-exploiting substitution techniques to determine Jacobian matrices. In this paper, we take a closer look at the underlying combinatorial problem. We propose a column ordering heuristic to augment the "usable sparsity" in the Jacobian matrix. Furthermore, we present a new elimination technique based on merging of successive columns.

Keywords: Sparse Jacobians, Substitution Methods, Partition, Merging

1 Introduction

Mathematical derivatives can be approximated or calculated by a variety of techniques including computational differentiation (CD) [6], finite differencing (FD) [7], and computational divided differencing (CDD) (or algorithmic differencing) [20]. In some of the CD tools, the sparsity information contained in the Jacobian matrix J is exploited by using the so called "sparse dynamic vector mode". The run-time penalty for the indirect access of data can be expensive [12]. Alternatively, if the sparsity pattern is known a priori or can be determined easily [13], then a "seed matrix" can be used to compress the Jacobian matrix. The Curtis, Powell, and Reid method, henceforth the CPR method, defines a seed matrix from a consistent partition of the columns of J. A partition of the columns of J is said to be consistent

*Email: hossain@cs.uleth.ca, WWW: http://www.cs.uleth.ca/~hossain
†Email: Trond.Steihaug@ii.uib.no, WWW: http://www.ii.uib.no/~trond

with the direct determination of J if no two columns in the same group (of columns in the partition) contain nonzeros in the same row position. Then the matrix J can be determined from the product JS where S is the seed matrix defined by the column partition.

Any of the aforementioned techniques can be used to determine the "compressed matrix" $A = JS$ from which the nonzeros can be obtained directly, by substitution, or by elimination. The computational cost of a determination procedure is expressed in terms of the number of AD passes (or extra function evaluations in FD) for computing A plus the cost of recovering the nonzeros of J from A. Therefore, fewer columns in S means fewer AD passes or function evaluations. In general,

$$numcol(S_{elim}) \leq numcol(S_{subs}) \leq numcol(S_{direct}) \tag{1}$$

where $numcol(S)$ denotes the number of columns in S. On the other hand, the relative recovery cost for the nonzeros is given by reversing the inequality (1).

Both Newsam and Ramsdell [17] and Geitner, Utke, and Griewank [9] use a "compressed matrix formulation" to estimate a sparse Jacobian matrix. Recently, we have shown that substitution schemes can be defined based on the sparsity pattern of the compressed Jacobian matrix [15]. The distribution of zeros in compressed Jacobian matrix rows determines how effectively this sparsity can be exploited. Let d_i be the longest sequence of consecutive zeros in row i of A. Then the substitution scheme saves

$$d = min_i \{d_i\}$$

AD passes or function evaluations. Thus d represents the "usable sparsity" in A i.e., how much of A's sparsity information can be exploited.

Although the compressed matrix rows may contain many zeros, the length (i.e., the number of zeros in the sequence) of the longest sequence of consecutive zeros may be much smaller than the number of zeros in the rows. Elimination methods are optimal in that number of AD passes needed is equal to the maximum number of nonzeros in any row of A. However, in addition to having to solve a general linear system of equations for each row of J, it is necessary to ensure that the computed values are sufficiently accurate. The main contributions of this paper are as follows.

1. Substitution
   We investigate ways to increase the length of consecutive zeros sequence. The general problem of increasing consecutive zeros is found to be hard. A column rearrangement heuristics is proposed.

2. Elimination
   We suggest a new elimination scheme based on successive column merging. Similar to the other compressed matrix techniques [9, 17], our method is optimal in terms of the number of AD passes. The nonzeros are solved for from banded systems for which efficient algorithms exist[11]. We also demonstrate that the numerical accuracy of the computed values is comparable to other elimination methods.

The remainder of the paper is organized as follows. In Section 2, we present the general matrix determination procedure and then briefly review the methods for obtaining the nonzeros.

In Section 3, we discuss our substitution scheme and introduce the problem of maximizing the usable sparsity in the compressed Jacobian. The general problem of maximizing consecutive zeros is NP-hard. We discuss a column rearrangement heuristic to improve the consecutive zeros in the compressed Jacobian.

Section 4 presents a new elimination procedure called "successive column merging". Our observation that the condition number of the reduced seed matrix grows polynomially in terms of the number of columns in the seed matrix promises practical utility of the method. Section 5 contains some numerical test results.

Finally, the paper is concluded in Section 6 with research directions currently under investigation.

In this paper if a capital letter is used to denote a matrix (e.g., A, B, X), then the corresponding lower case letter with subscript ij refers to the (i,j) entry (e.g., a_{ij}, b_{ij}, x_{ij}). Whenever appropriate, we also use colon notation [11] to specify a submatrix of a matrix. For example, for $A \in R^{m\times n}$, $A(i,:)$ and $A(:,j)$ designates ith row and jth column respectively. And for a vector of indices $v, A(:,v)$ will be the submatrix consisting of the columns with indices in v.

2 Techniques for calculating derivatives

Let $A \in R^{m\times p}$ be the matrix we want to determine. Denote by ρ_i the number of nonzeros in $A(i,:)$ and let $\rho = \max_i \rho_i$ be such that $\rho \le q \le p$ for some positive integer q. Let v_i be the vector of indices of nonzero elements in $A(i,:)$. Without loss of generality we assume that $\rho_i = \rho$. Suppose $S \in R^{p\times q}$ be a given seed matrix. Then the following procedure can be used to determine A

1. Obtain $B = AS$ as q matrix-vector products (using CD or FD or CDD).
2. Identify the reduced seed matrix $\hat{S}_i \in R^{q\times q}$ corresponding to $A(i,v_i)$ (the nonzeros of $A(i,:)$)
$$\hat{S}_i = S(v_i,:).$$
3. Solve for unknown elements $a_{ik} \ne 0$ of $A(i,:)$
$$\hat{S}_i^T A(i,v_i)^T = B(i,:)^T.$$

Suppose $F'(x_0) \equiv J$ be the Jacobian matrix of the mapping $F: R^m \mapsto R^n$ evaluated at a given point x_0. The columns of J can be partitioned such that columns in the same group are structurally orthogonal. Columns in the same group in a (consistent)partition can be determined together. A consistent column partition can be described by a $0-1$ matrix $X \in R^{n\times p}$. Then, $JX = A$ is the "compressed Jacobian matrix" corresponding to partition X. This would mean that matrix S in the above procedure represents a second compression of J. Note that each row i of J and A has the same number of nonzeros ρ_i. For the purpose of our presentation, the distinction between the Jacobian matrix J and an associated compressed Jacobian matrix A is made only as necessary.

Depending on the structure of S, the nonzeros of A can be recovered from B directly, by a substitution procedure, or by an elimination (solution of a general linear system of equations) procedure.

To illustrate we consider the matrix
$$A = \begin{bmatrix} a_{11} & 0 & a_{13} \\ a_{21} & a_{22} & 0 \\ 0 & a_{32} & a_{33} \end{bmatrix}.$$

2.1 Direct methods

The direct determination of A consists of reading off the nonzeros from the matrix B without any arithmetic operations. It can be shown that each row of S is the kth coordinate vector e_k for some k, a vector whose kth element is 1 and the remaining elements are zeros and, $\hat{S}_i$ are identity matrices of appropriate dimension[1]. The CPR and the Coleman-Moré [3] methods are direct methods.

2.2 Substitution methods

In a substitution method, there is an ordering of the nonzeros of A such that the nonzeros can be determined using the previously computed values. In other words, the reduced seed matrix $\hat{S}_i$ corresponding to the ith row of A is triangular (or can be permuted to a triangular matrix).

Let the seed matrix be

$$S = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Then, $\hat{S}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ and the system to be solved for the nonzeros in row 2 of A in step 3 of the above procedure is

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} b_{21} \\ b_{22} \end{bmatrix}.$$

It is clear that $\hat{S}_1$ and $\hat{S}_3$ are also triangular.

Most work on substitution methods are concerned with the determination of sparse Hessian matrices [1, 4, 10, 19]. Hossain and Steihaug [15], Coleman and Verma [5], and Plassmann [18] discuss substitution techniques for the determination of sparse Jacobian matrices.

2.3 Elimination methods

These are general methods where no special structure is assumed for the seed matrix. Any $q \times q$ submatrix of S must, however, be nonsingular. The ideal seed matrix would satisfy the properties that any square submatrix is well conditioned and easy to solve. In Newsam and Ramsdell, [17] two family of seed matrices have been considered.

Recall that if

$$p(\lambda) = \sum_{j=1}^{q} a_j \lambda^{j-1}$$

is a polynomial of degree $q-1$ where the coefficients a_j are to be determined by interpolation, then $p(\lambda_j) = p_j$ for $j = 1, \ldots, q$ defines a system of q linear equations from which the coefficient vector $a \in R^q$ is uniquely determined if λ_i are distinct. Thus the set

$$\left\{ \lambda^{j-1} : \lambda \in R \text{ for } j = 1, \ldots, q \right\}$$

[1]With FD, $\hat{S}_i$ are diagonal.

forms a basis for the function space spanned by polynomials. In matrix notation, the linear system is written

$$V^T a = p$$

where V is the so called Vandermonde matrix. The Vandermonde system can be solved in $O(q^2)$ floating point operations [11]. The difficulty is that the numerical conditioning of the system deteriorates exponentially with the size of V. This difficulty with conditioning is improved by choosing Chebychev polynomials T_i that forms an orthogonal basis for the polynomial function space. Unlike the Vandermonde system, however, solvers for the Chebychev polynomial based system need $O(q^3)$ floating point operations. In Section 4, we present a new elimination method that can be performed in linear time and the conditioning of the system compares favorably with the Vandermonde system.

3 Computation by substitution

Let $C_k, k = 1, \ldots, p$ be a set of column indices defining a consistent partition of the columns of J. Then the seed matrix $X \in R^{m \times p}$ corresponding to the partition is defined as

$$x_{ik} = \begin{cases} 1 & \text{if column index } i \text{ belongs to } C_k \\ 0 & \text{otherwise} \end{cases}$$

The observation that reduced seed matrices corresponding to the rows of J are triangular is characterized by the following result which summarizes the results in [15].

Lemma 1 *If $p > \rho$ and each row of $JX \equiv A \in R^{m \times p}$ has at least d consecutive zero entries with $d \leq p - \rho$, then there is a seed matrix $S \in R^{m \times (p-d)}$ defined as*

$$S(:, i) = \sum_{j=0}^{d} X(:, i + j), i = 1, 2, \ldots, p - d$$

such that each $\hat{S}_i, i = 1, 2, \ldots, m$ corresponding to row i of A (and hence J) are triangular and nonsingular.

For $d = 1$, we have a substitution procedure that saves at least one AD pass or an extra function evaluation. A result similar to our result when $d = 1$ is reported in Plassman [18] although no algorithm for constructing a seed matrix is given.

In view of Lemma 1, it is clear that if A contains many zeros in rows but the zeros are scattered in smaller sequences, the sparsity can not be exploited adequately.

Let d_i be defined as in Page 1. The occurrence of zero sequences in the compressed matrix rows determines the usable sparsity

$$d = min_i \{d_i\}.$$

Let

$$A = \begin{bmatrix} a_{11} & 0 & a_{13} & a_{14} & 0 \\ 0 & a_{22} & a_{23} & 0 & a_{25} \\ a_{31} & a_{32} & 0 & a_{34} & 0 \end{bmatrix}$$

be the matrix we want to determine. Then $d_1 = d_2 = d_3 = 1 = d$. Although each row contains two zeros but they are not usable. The associated seed matrix is

$$S = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

We can address this problem as rearrangement of the columns of A. We want to find a permutation of the columns of A so that all three rows of AP have 2 consecutive zeros. For example, a suitable permutation vector is $\begin{bmatrix} 1 & 4 & 3 & 5 & 2 \end{bmatrix}$. The general permutation problem can be stated as follows:

The Column Permutation Problem (CPP)

Given an $m \times p$ matrix A and a positive integer d, is there a permutation matrix P such that each row of AP will have at least d consecutive zeros?

Unfortunately, the CPP problem is hard. A transformation from the SATISFIABILITY(SAT) problem [16] to CPP can be constructed to show that the problem is NP-complete. Omitting details, the central idea in the transformation is that for each variable and each clause of an arbitrary instance of the SAT problem we define a column of a $0-1$-matrix A. The arrangement of 1's would be such that the SAT instance is satisfiable if and only if each row A has d consecutive zeros.

Theorem 1 *The CPP problem is NP-complete for $d \geq 3$*

We recall that the matrix A is the compressed Jacobian matrix obtained as JX where X is the matrix defined by a consistent partition of the columns of J. We may state the general problem of determination of a sparse Jacobian matrix by substitution as

Given a Jacobian matrix $J \in R^{m\times n}$ and a positive integer d, find $0-1$ matrices X, S, and P of appropriate dimension such that J can be determined from the product $JXPS$ where

- *the matrix X defines a consistent column partition and*
- *each row of JXP has at least d consecutive zeros.*

The problem of obtaining optimal consistent column partition is already known to be NP-hard [3, 21]. So, practical approaches to solving this problem will most likely be directed toward designing good heuristics.

The central observation to improving the usable sparsity is to recognize that the matrix A is obtained as $A = JX$. To improve usable sparsity we consider the following heuristics:

- Rearrange the column indices between the groups defining a consistent partition of J.
- Permute the columns of A.

For the rearrangement of the columns of J in the partition we use a greedy approach that tries to increase the length of the sequences of consecutive zeros.

Let A be the compressed matrix and the maximum number of consecutive zeros d in A has been computed. Locate the rows called *probrows* where all the consecutive zero sequences are of length d or less and at least one of the zero sequences has length d (for non *probrows* the longest sequence of zero elements must be more than d). For example, if one such *probrows* has the following pattern

$$\begin{array}{cccccccccccccccc} [& \ldots & \times & 0 & 0 & \times & 0 & 0 & \times & \times & \times & 0 & 0 & \times & \times & \ldots &] \\ & & & & & \uparrow & & & & & & & & & & & \end{array}$$

(where $\times$ denotes a nonzero element) and $d = 2$, we try to include the column of J corresponding to the nonzero pointed by $\uparrow$ in a different column of A (if successful this will increase the value of d by more than one in this case). If there is a column of A where the new column can be inserted then we are done. In order to insert a column in a group, besides being structurally orthogonal, it must be ensured that the nonzeros in other row positions of the column satisfy the following:

1. If it is not in a *probrows*, it must not decrease the number of consecutive zeros in that row.
2. If it is in a *probrows* (and not the one that we are currently processing), it must not decrease the value of d for that row.

We continue the above procedure for each of the *probrows*. If successful, we have the value of d incremented by 1. The process is repeated on the new compressed Jacobian as long as there are improvements in d.

With a greedy approach such as the CPR to obtain the column partitioning, it is expected that the the lower numbered columns of A will contain most of the columns of J. Therefore, these columns of A are likely to be denser than the higher numbered columns . This suggests that the zero-sequences of length d be considered from the left so that a column of J that has a nonzero in the *probrows* has a better chance of inclusion without conflict in a higher numbered column of A.

4 Computation by elimination

For a general elimination scheme, a seed matrix in which every square submatrix is well conditioned and easy to solve seems to be rather conflicting requirements. The "successive column merging" technique that we propose here allows a flexible solution where, depending on the accuracy requirements in the computed values, the linear solver can be made efficient.

Let

$$A = \begin{bmatrix} a_{11} & 0 & a_{13} & a_{14} & 0 \\ 0 & a_{22} & a_{23} & 0 & a_{25} \\ a_{31} & a_{32} & 0 & a_{34} & 0 \end{bmatrix}$$

be a compressed Jacobian matrix. We perform a merge of the neighboring columns of A to obtain

$$A' = \begin{bmatrix} a_{11} & a_{13} & a_{13} + a_{14} & a_{14} \\ a_{22} & a_{22} + a_{23} & a_{23} & a_{25} \\ a_{31} + a_{32} & a_{32} & a_{34} & a_{34} \end{bmatrix}$$

and a second merge gives

$$A'' = \begin{bmatrix} a_{11} + a_{13} & 2a_{13} + a_{14} & a_{13} + 2a_{14} \\ 2a_{22} + a_{23} & a_{22} + 2a_{23} & a_{23} + a_{25} \\ a_{31} + 2a_{32} & a_{32} + a_{34} & 2a_{34} \end{bmatrix}$$

Then the seed matrix corresponding to the merged compressed matrix is

$$S = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}.$$

Let B be computed as in Section 2. Then solving for the unknowns in row 1 of A will be

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{13} \\ a_{14} \end{bmatrix} = \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \end{bmatrix}.$$

A MATLAB [22] experiment gives $4.4243 \leq cond(\hat{S}_i) \leq 13.9312$ where $cond(\hat{S}_i)$ is the MATLAB 2-norm condition number estimate taken over reduced square submatrices $\hat{S}_i$ of S.

Interestingly, the seed matrix defined by the merging procedure can be described by the famous Pascal's triangle. The dth row of Pascal's triangle is defined by the coefficients of terms in the expansion for $(a+b)^d$ where d is a nonnegative integer and $a, b \in R$. For example, the 3rd row of the Pascal's triangle is $1 \quad 3 \quad 3 \quad 1$.

Let O_j denote the zero vector in R^j. Then column i in the seed matrix S is (row i in S^T)

$$[O_{i-1} \; u \; O_{p-d-i}]$$

where component j in $u \in R^{d+1}$ is the binomial coefficient $\begin{pmatrix} d \\ j-1 \end{pmatrix}$. The discussion is summarized in the following result.

Denote the maximum number of nonzeros in any row of A by ρ.

Theorem 2 *Given a compressed Jacobian matrix $A \in R^{m \times p}$ and a positive integer $d \leq p - \rho$, the reduced seed matrices $\hat{S}_i$ of the seed matrix S obtained by d column merges, are nonsingular banded matrices of bandwidth $d+1$.*

That the seed matrix is constructed from Pascal's triangle makes it convenient for the storage and manipulation of the reduced seed matrices. To solve for the nonzeros in each row of A, we can use a band LU factorization where the band structure is retained in the computed triangular factors. Then a band forward substitution procedure can be used to obtain the solution efficiently. Therefore, the cost for the solver is approximately $2qd^2$ floating point operations for the band LU factorization and $2qd$ floating point operations for the band triangular system solve giving an $O(qd^2)$ algorithm. For $d \ll q$, the asymptotic time complexity for successive column merging procedure is therefore a linear function of q which is very efficient compared with the quadratic complexity $O(q^2)$ for Vandermonde solvers and cubic complexity $O(q^3)$ for the Chebychev polynomial based system.

5 Numerical experiments

In this section, we describe the computational experiments for the procedures described in sections 3 and 4. The test pattern matrices are drawn from the Harwell-Boeing test collections [8] and randomly generated pattern matrices. A consistent

		Direct(DSM)		Substitution
	ρ	mngp	p	d
Total	200	217	223	11

Table 1: Substitution results for SMTAPE collection

column partitioning is obtained by using DSM [2]. Table 1 shows results from SMTAPE matrix collection. The number of groups in partition (and hence the number of columns in the compressed matrix) is denoted by p, $mngp$ is a DSM computed lower bound on the number of groups, and d is the number of AD passes saved by substitution without column rearrangement. The total is taken over matrices for which $p > \rho$. Comparing $p - d$ with $mngp$ we conclude that the substitution scheme is more efficient than the optimal direct determination. The second set of test pattern matrices are from the collection CHEMIMP. We test the column interchange algorithm described in Section 3. The results are given in Table 2. The value of d is specified before and after the column interchange has been done. Although the improvements are not spectacular, they do provide directions for further investigation.

	mngp	p	d(before)	d(after)
impcola	8	8	0	0
impcolb	10	11	2	3
impcolc	8	8	0	0
impcold	10	11	1	1
impcole	20	21	1	2
Total	56	59	4	6

Table 2: Results for CHEMIMP: Chemical engineering plant models

	density(%)	$mngp$	p	d(before)	d(after)
1.	1.0%	12.6	12.8	1.0	1.2
2.	2.0%	20.2	29.0	2.0	2.8
3.	3.0%	27.6	49.8	3.8	4.2
4.	4.0%	34.4	75.8	5.0	5.4
Total		94.8	167.4	11.8	13.8

Table 3: Average numbers for 500×500 random matrices

Finally, we test the column interchange algorithm on randomly generated sparsity pattern and the results are shown in Table 3. Sets of random pattern matrices of different densities are generated using MATLAB `sprandn` function. The results shown are the averages for each set. For many of the patterns, the DSM partition is optimal. Again, the column interchange alone is not as effective for reducing the value of d. Nevertheless, combined with column permutation mentioned in 3, column interchange heuristics is expected to perform better.

Our next experiments are concerned with the seed matrices obtained from column merging. The experiments have been conducted in MATLAB. For Vandermonde matrices we choose [12]

$$\lambda_j = 2\frac{(j-1)}{(q-1)} - 1 \text{ for } j = 1, 2, \ldots, q$$

	Column Merging				Vandermonde			
d	p =10	20	30	50	10	20	30	50
4	1.17e+03	3.61e+04	2.156e+05	1.878e+06	2.33e+03	1.16e+08	6.17e+12	1.90e+19
5	2.101e+03	2.179e+05	2.161e+06	3.370e+07	1.30e+03	1.19e+08	5.35e+12	2.63e+19

Table 4: Condition estimate for seed matrices

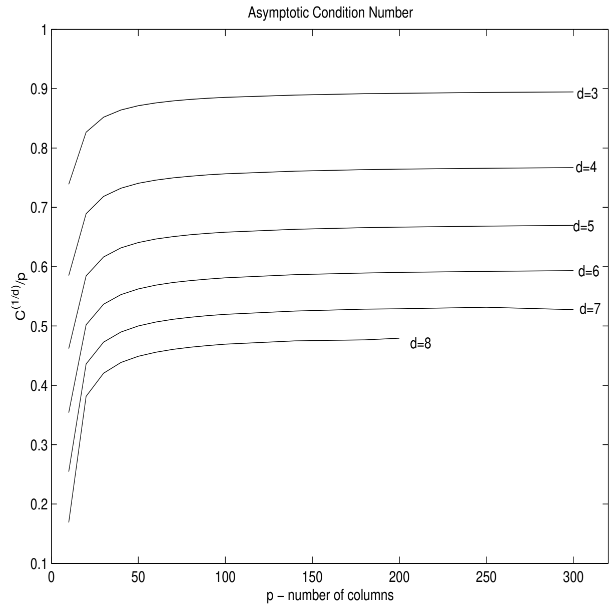

Figure 1: Condition estimate for seed matrices

Table 4 contains spectral condition number estimates for Vandermonde and column merging reduced seed matrices. Here, d denotes minimum number of zeros in any row of $A \in R^{m \times p}$. We estimate spectral condition numbers for $p = 10, 20, 30, 50$ for the two types of seed matrices. For the column merging seed matrices, we list the condition number estimates for the reduced seed matrices with highest condition number for each p. It is quite clear that the numerical conditioning of the column merging seed matrices are superior to Vandermonde matrices. Finally, we illustrate that the relationship between the growth of reduced seed matrix condition number and the choice of p and d leads to a flexible solution in terms of accuracy of the solution and computational cost of the procedure. Let $C_d(p)$ be the largest condition number of any reduced seed matrix of the (Pascal triangle) seed matrix with d zeros and p columns. Figure 1 plots $C_d(p)^{1/d}/p$ for $d = 3, 4, 5, 6, 7, 8$ and $p \leq 400$. As can be seen from the figure, $C_d(p)^{1/d}/p$ reaches stationary value very rapidly and we can conclude that $C_d(p) = c_d p^d$ for $d = 3, 4, 5, 6, 7, 8$ and $p \leq 400$ where c_d is independent of p and $c_d < 1$ for $d \geq 3$. Hence, the condition number grows with p as p^d. If we accept a condition number of 10^8 and we want to utilize the fact that every row has 4 zeros, we can allow as many as $p = 100$. Very few of the standard test problems have that many columns in the compressed Jacobian matrix.

6 Concluding remarks

The goal of this paper was to explore the possibility of different seed matrices for the efficient determination of Jacobian matrices. Our preliminary numerical results with column merging and column rearrangement heuristic seed matrices are encouraging and should motivate further study. Currently we are in the process of implementing an optimization formulation of the column permutation problem. Further, we believe that a graph theoretic view point may provide more insights into the problem. This is also under investigation.

7 Acknowledgments

The work of the first author was supported by the Natural Sciences and Engineering Research Council of Canada under RGPIN and the work of the second author was supported by the Research Council of Norway.

The authors would like to thank Jiri Fiala for the communication on proof idea of Theorem 1.

References

[1] T. F. Coleman and J.-Y. Cai. The cyclic coloring problem and estimation of sparse Hessian matrices. *SIAM J. Alg. Disc. Meth.*, 7(2):221–235, 1986.

[2] T. F. Coleman, B. S. Garbow, and J. J. Moré. Software for estimating sparse Jacobian matrices. *ACM Trans. Math. Software*, 10(3):329–345, 1984.

[3] T. F. Coleman and J. J. Moré. Estimation of sparse Jacobian matrices and graph coloring problems. *SIAM J. Numer. Anal.*, 20(1):187–209, 1983.

[4] T. F. Coleman and J. J. Moré. Estimation of sparse Hessian matrices and graph coloring problems. *Math. Programming*, 28:243–270, 1984.

[5] T. F. Coleman and A. Verma. The efficient computation of sparse Jacobian matrices using automatic differentiation. *SIAM J. Sci. Comput.*, 19(4):1210–1233, 1998.

[6] G. Corliss, C. Faure, A. Griewank, L. Hascoët, and U. Naumann, editors. *Automatic Differentiation: From Simulation to Optimization.* Computer and Information Science. Springer, New York, 2002.

[7] A. R. Curtis, M. J. D. Powell, and J. K. Reid. On the estimation of sparse Jacobian matrices. *J. Inst. Math. Appl.*, 13:117–119, 1974.

[8] I. S. Duff, R. G. Grimes, and J. G. Lewis. Users' guide for the Harwell-Boeing sparse matrix collection (Release I). Technical Report tr/pa/92/86, CERFACS, 1992.

[9] U. Geitner, J. Utke, and A. Griewank. Automatic computation of sparse Jacobians by applying the method of Newsam and Ramsdell. In M. Berz, C. Bischof, G. Corliss, and A. Griewank, editors, *Computational Differentiation: Techniques Applications, and Tools*, pages 161–172. SIAM, Philadelphia, Penn., 1996.

[10] D. Goldfarb and P. L. Toint. Optimal estimation of Jacobian and Hessian matrices that arise in finite difference calculations. *Mathematics of Computation*, 43(167):69–88, 1984.

[11] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

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

[13] A. Griewank and C. Mitev. Detecting Jacobian sparsity patterns by Bayesian probing. Preprint IOKOMO-04-2000, Technical University of Dresden, 2000. Submitted to Math. Progr.

[14] A. Griewank and C. Mitev. Verifying jacobian sparsity. In Corliss et al. [6], chapter 32, pages 271–279.

[15] S. Hossain and T. Steihaug. Reducing the number of AD passes for computing a sparse Jacobian matrix. In Corliss et al. [6], chapter 31, pages 263–270.

[16] M.R.Garey and D.S.Johnson. *Computers and Intractibility*. W.H.Freeman, San Francisco, 1979.

[17] G. N. Newsam and J. D. Ramsdell. Estimation of sparse Jacobian matrices. *SIAM J. Alg. Disc. Meth.*, 4(3):404–417, 1983.

[18] P. E. Plassmann. Sparse Jacobian estimation and factorization on a multiprocessor. In T. F. Coleman and Y. Li, editors, *Large-Scale Optimization*, pages 152–179. SIAM, Philadelphia, Penn., 1990.

[19] M. J. D. Powell and P. L. Toint. On the estimation of sparse Hessian matrices. *SIAM J. Numer. Anal.*, 16:1060–1074, 1979.

[20] L. B. Rall and T. W. Reps. Algorithmic differencing. In A. Facius and U. Kulisch, editors, *Perspectives on Enclosure Methods*. Springer-Verlag, Vienna, 2001.

[21] T. Steihaug and A. S. Hossain. Graph coloring and the estimation of sparse Jacobian matrices with segmented columns. Technical Report 72, Department of Informatics, University of Bergen, 1997.

[22] The MathWorks. Matlab 5.3.1, 2000. See www.mathworks.com/products/matlab.