Graph Coloring in Optimization Revisited

Assefaw Hadish Gebremedhin *   Fredrik Manne †
Alex Pothen

February 20, 2004

Abstract

Algorithms for solving numerical optimization problems and nonlinear systems of equations that rely on derivative information require the repeated estimation of Jacobian and Hessian matrices. The problem of minimizing the number of function evaluations needed to estimate the derivative matrices, within a finite difference method or an automatic differentiation method, has been formulated as a collection of (specialized) graph coloring problems. These formulations have led to the successful use of graph coloring algorithms in optimization software.

We survey the historical development of graph coloring formulations and algorithms in the context of estimating derivatives in optimization. (We believe this is the first such survey.) A dozen variations of derivative matrix estimation problems exist, depending on the matrix to be evaluated, whether all or only a subset of the nonzero matrix elements need to be evaluated, and on the details of the estimation techniques. The plurality of problems has led to a plethora of techniques, and the inherent similarity of the various estimation problems has thus far been obscured. This fragmentation also makes it difficult to identify a generic formulation, thereby hindering the development of algorithms and flexible software. Given the incremental nature of advances in research, this fragmentation was perhaps inevitable.

With the advantages of about twenty years of hindsight, we describe a unified graph theoretic framework for solving the various matrix estimation problems, based on a graph-theoretic characterization

*Computer Science Department, Old Dominion University, Norfolk, VA 23529-0162 USA. {assefaw, pothen}@cs.odu.edu

†Department of Informatics, University of Bergen, N-5020 Bergen, Norway. fredrikm@ii.uib.no

of structural orthogonality. This integrative framework provides additional insights into existing algorithms, enables the design of several new algorithms, and facilitates the solution of new problems. We report computational results for two representative problems to demonstrate the advantages of the new algorithms. We outline several open problems that result from the work described here.

The use of graph coloring in numerical optimization is one of the problems in the emerging new research area of combinatorial scientific computing, a research area where combinatorial mathematics and algorithms are used to solve problems in scientific computing. Our broader hope is that this survey provides an impetus for the formulation and solution of new combinatorial problems in scientific computing.

Key words: Sparsity, symmetry, Jacobians, Hessians, finite differences, automatic differentiation, matrix partitioning problems, graph coloring problems, NP-completeness, approximation algorithms

Contents

1 Introduction

2 Definitions of Graph Theoretic Concepts

2.1 Preliminaries

A *graph* G is an ordered pair (V, E) where V is a finite and nonempty set of *vertices* and E is a set of unordered pairs of distinct vertices called *edges*. If $(u, v) \in E$, vertices u and v are said to be *adjacent*; otherwise they are called *non-adjacent*. A *path* of *length* l in a graph is a sequence $v_1, v_2, \ldots, v_{l+1}$ of distinct vertices such that v_i is adjacent to v_{i+1}, for $1 \leq i \leq l$. Two distinct vertices are said to be *distance-k neighbors* if the shortest path connecting them has length *at most* k; otherwise they are called *non-distance-k neighbors*. The number of distance-k neighbors of a vertex u is referred to as the *degree-k* of u.

In a graph $G = (V, E)$, a set of vertices $C \subseteq V$ is said to *cover* a set of edges $F \subseteq E$ if for every edge $e \in F$, at least one of the endpoints of e is in C. If the set C covers the entire E, it is called a *vertex cover*. A set of vertices $I \subseteq V$ is called an *independent set* if no pair of vertices in I are adjacent to each other. A set $Q \subseteq V$ is called a *clique* if the vertices in Q are mutually adjacent to each other.

A graph $G = (V, E)$ is *bipartite* if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in E connects a vertex from V_1 to a vertex from V_2.

Notations. We introduce some notations used in the rest of this paper. Let $N_k(u) = \{w : w \text{ is a distance-}k \text{ neighbor of } u\}$. Let $d_k(u) = |N_k(u)|$ denote the degree-k of u; Δ be the maximum degree-1 in G; and $\overline{\delta}_k = \frac{1}{|V|}\sum_{u \in V} d_k(u)$ denote the average degree-k in G. Further, in a bipartite graph $G_b = (V_1, V_2, E)$, let the maximum degree-1 in the vertex sets V_1 and V_2 be denoted by $\Delta(V_1)$ and $\Delta(V_2)$, respectively. Similarly, let the average degree-k in the sets V_1 and V_2 be denoted by $\overline{\delta}_k(V_1)$ and $\overline{\delta}_k(V_2)$, respectively.

2.2 Distance-k Graph Coloring

A *distance-k p-coloring*, or (k, p)-coloring for short, of a graph $G = (V, E)$ is a mapping $\phi : V \to \{1, 2, \ldots, p\}$ such that $\phi(u) \neq \phi(v)$ whenever u and v are distance-k neighbors. The minimum possible value of p in a (k, p)-coloring of a graph G is called its *k-chromatic number*, and is denoted by $\chi_k(G)$. A (k, p)-coloring of $G = (V, E)$ is called *partial* if it involves only a subset of the vertices; in particular, a partial (k, p)-coloring of $G = (V, E)$ on W, $W \subset V$, is a mapping $\phi : W \to \{1, 2, \ldots, p\}$ such that $\phi(u) \neq \phi(v)$ whenever u and v are distance-k neighbors.

2.3 Representing Matrix Structures Using Graphs

Bipartite graph Let A be an $m \times n$ matrix with rows $r_1, r_2, \ldots, r_m$ and columns $a_1, a_2, \ldots, a_n$. We define the bipartite graph $G_b(A)$ of A as $G_b(A) = (V_1, V_2, E)$ where $V_1 = \{r_1, r_2, \ldots, r_m\}$, $V_2 = \{a_1, a_2, \ldots, a_n\}$, and $(r_i, a_j) \in E$ whenever a_{ij} is a nonzero element of A, for $1 \leq i \leq m$, $1 \leq j \leq n$.

Note that $G_b(A)$ is a space-efficient representation for a nonsymmetric matrix A. To see this, notice that the number of vertices $|V_1| + |V_2| = m + n$, and the number of edges $|E| = nnz(A)$, where $nnz(A)$ is the number of nonzeros in A. Also, note that the graph can be constructed by reading off the entries of the matrix without any further computation.

Adjacency graph Let $A \in R^{n \times n}$ be a symmetric matrix with nonzero diagonal elements and let its columns be $a_1, a_2, \ldots, a_n$. The adjacency graph of A is $G(A) = (V, E)$ where $V = \{a_1, a_2, \ldots, a_n\}$, and $(a_i, a_j) \in E$ whenever a_{ij} is a nonzero element of A, for $1 \leq i \leq n$, $1 \leq j \leq n$, $i \neq j$. Note that $G(A)$ is a space-efficient, symmetry-exploiting, graph representation of the symmetric matrix A with no explicit representation for the edges corresponding to the nonzero diagonal elements. In particular, the number of vertices is n and the number of edges is $\frac{1}{2}(nnz(A) - n)$. This is in contrast to $2n$ and $nnz(A)$, respectively, had a bipartite graph representation been used.

Column intersection graph Our graph formulations for the derivative matrix estimation problems considered in this paper are based solely on the two graph representations discussed above. However, in the literature, other graph representations have also been used. In particular, Coleman and Moré [7] used the column intersection to represent a matrix.

Let $a_1, a_2, \ldots, a_n$ correspond to the columns of an $m \times n$ matrix A such that each $a_i = \{j \in \{1, \ldots, m\} : a_{ij} \neq 0\}$. The column intersection graph of A is $G_c = (V, E)$ where $V = \{a_1, \ldots, a_n\}$ and $(a_i, a_j) \in E \iff a_i \bigcap a_j \neq \emptyset$. In other words, an edge (a_i, a_j) exists whenever columns a_i and a_j have nonzero entries at at least one common row position.

In the rest of the pape, in order to simplify notation, we may use a_i both in reference to the ith column of matrix A and the corresponding vertex in an appropriate graph of A.

3 Direct Estimation of Jacobians Using Unidirectional Partition

Given a continuously differentiable function $F : R^n \to R^m$, the **Jacobian** of F at the point x is the $m \times n$ matrix whose (i, j) entry $J(x)_{ij} = F'(x)_{ij} = \frac{\partial f_i}{\partial x_j}(x)$, where $f_1(x), f_2(x), \ldots, f_m(x)$ are the components of $F(x)$. Let A denote the Jacobian matrix $F'(x)$. An estimate for the jth column of A, denoted henceforth by a_j, can be obtained from the *forward difference* approximation,

$$Ae_j = a_j = \frac{\partial}{\partial x_j} F(x) \approx \frac{1}{h}[F(x + he_j) - F(x)], \quad 1 \leq j \leq n, \tag{1}$$

where e_j is the jth unit vector and h is a positive step length. Other finite difference (FD) approximations of higher order, such as *central differences*, could also be used to estimate A. In any case, if $F(x)$ is already evaluated, an approximation to a_j is obtained with one additional function evaluation. Thus, if each column of A is computed independently, n additional function evaluations will be required. However, by exploiting the sparsity structure of A, the required number of function evaluations can be reduced significantly. The sparsity structure of A is often easily available and the goal here is to exploit this to estimate the nonzero entries of A using as few function evaluations as possible under the assumption that evaluating $F(x)$ is more efficient than evaluating the components $f_i(x)$, $1 \leq i \leq m$, separately.

Let d_i be a binary vector obtained by adding some unit vectors e_j, $j \in \{1, 2, \ldots, n\}$, together. The problem of estimating a sparse Jacobian matrix using FD can then be stated as follows. Given the sparsity structure of matrix A find binary vectors $d_1, d_2, \ldots, d_p$ such that the products $Ad_1, Ad_2, \ldots, Ad_p$ enable the determination of all the nonzero entries of A.

Specifying the vectors $Ad_1, Ad_2, \ldots, Ad_p$ gives rise to a system of linear equations where the unknowns are the nonzero elements of A. If the choice of the vectors d_i is such that the resulting system of equations can be ordered to a diagonal form, then we say that A is *directly* determined by the vectors d_i. If, on the other hand, the vectors d_i are chosen such that the system of equations can be ordered to a triangular form, then the unknowns can be determined via *substitution*.

In both a direct and a substitution based determination, minimizing the number of function evaluations corresponds to minimizing the number of vectors p. There is, however, a trade-off in the choice of methods. A direct method is more restrictive and hence requires more function evaluations compared to a substitution method. On the other hand, a substitution method is subject to numerical instability, whereas a direct method is not. Moreover, in terms of parallel computation, direct methods offer straightforward parallelization since the estimates can be read off directly from each row of a matrix-vector product, whereas substitution methods have less parallelism since there are more dependencies among the computations required to obtain the matrix entries. In the current and the next two sections, we consider direct methods; substitution methods will be discussed in Section 6.

3.1 The Matrix Partitioning Problem

In a direct determination of a matrix A, note that for each nonzero element a_{ij}, there is some vector d_k in the set $\{d_1, d_2, \ldots, d_p\}$ such that $a_{ij} = (Ad_k)_i$, where $(Ad_k)_i$ is the ith component of the vector Ad_k. Thus, each nonzero matrix element a_{ij} can be read off from a component of some vector Ad_k.

One way of stating the problem that arises in the direct, efficient estimation of a sparse Jacobian matrix is as follows.

Problem 3.1 *Given the sparsity structure of a general matrix $A \in R^{m\times n}$, find the fewest binary vectors $d_1, d_2, \ldots, d_p$ such that $Ad_1, Ad_2, \ldots, Ad_p$ determine A directly.*

Curtis, Powell and Reid [12] were the first to observe that, while using a direct method, a group of columns can be determined by one evaluation of Ad if no two columns in this group have a nonzero in the same row position. Such columns are *structurally orthogonal*, since their pairwise inner products are zero. Based on this observation, Curtis, Powell and Reid suggested an algorithm for partitioning the columns of a matrix into a small number of groups consisting of structurally orthogonal columns. Their method, known often as the CPR technique, laid the foundation for a number of subsequent studies, including the celebrated work of Coleman and Morè [7]. In [7], the notion of *consistent partition* was introduced to formalize the CPR technique.

Definition 3.2 A partition of the columns of a matrix A is said to be *consistent* with a direct determination of A if whenever a_{ij} is a nonzero element of A then the group containing a_j has no other column with a nonzero in row i.

Figure 1 shows the sparsity structure of a matrix and a consistent partition of its columns. In the figure columns of the same color belong to the same group of the partition.

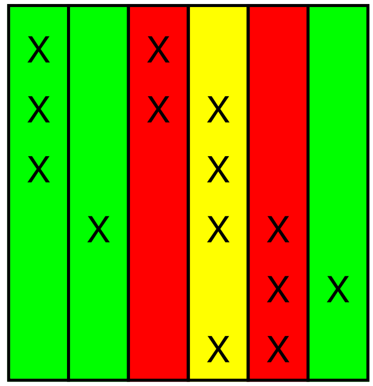


Figure 1: A consistent column partition

Let $\{C_1, C_2, \ldots, C_p\}$ be a consistent partition. With each group C_k, associate a binary vector d_k having components $\delta_j = 1$ if a_j belongs to C_k, and $\delta_j = 0$ otherwise. Then,

$$Ad_k = \sum_{\forall a_j} \delta_j a_j = \sum_{a_j \in C_k} a_j.$$

If $a_{ij} \neq 0$ and column $a_j \in C_k$, then $a_{ij} = (Ad_k)_i$. Thus, all the nonzero entries of A can be determined with p evaluations of Ad_k.

Using Definition 3.2, Problem 3.1 can be stated as a partitioning problem in the following way. In the remainder of this paper, we shall use the acronym MPP to refer to a matrix partitioning problem. MPP1 is the first of the eight problems considered in this paper.

Problem 3.3 (*MPP1*) *Given the sparsity structure of a matrix $A \in R^{m \times n}$, find a consistent partition of the columns of A that has the least number of groups.*

3.2 The Graph Formulation

Recall that a distance-k p-coloring (or (k,p)-coloring) of a graph $G = (V, E)$ is a mapping $\phi : V \to \{1, 2, \ldots, p\}$ such that $\phi(u) \neq \phi(v)$ whenever u and v are distance-k neighbors.

The *distance-k graph coloring problem* (DkGCP) asks for an optimal (k,p)-coloring of a graph: given a graph G and an integer k, find a (k,p)-coloring of G such that p is minimized.

Notice that a (k,p)-coloring of $G = (V, E)$ partitions the set V into p groups (called *color classes*) $U_1, U_2, \ldots, U_p$, where $U_i = \{u \in V : \phi(u) = i\}$. Each color class is a *distance-k independent set*, i.e., no pair of distinct vertices consists of distance-k neighbors. This prompts a natural question–does there exis a graph representation of a matrix structure such that the partitioning problem MPP1 be related to the DkGCP for some value of k?

Recall that structural orthogonality is the criterion used in a consistent partition of the columns (or rows) of a matrix. The following simple observation provides a graph theoretic characterization of structural orthogonality.

Lemma 3.4 *Let $A \in R^{m \times n}$ be a matrix and $G_b(A) = (V_1, V_2, E)$ be its bipartite graph. Two columns (or rows) of A are structurally orthogonal if and only if the corresponding vertices in $G_b(A)$ are non-distance-2 neighbors.*

Proof: We prove the statement for columns; a similar argument can be used to prove the case for rows. Assume that vertices a_i and a_j in V_2 are non-distance-2 neighbors. Thus, by definition, there is no path a_i, r_k, a_j in G_b for any $r_k \in V_1$, $1 \leq k \leq m$. In terms of matrix A, this means that there is no $k \in [1, m]$ such that both a_{ki} and a_{kj} are nonzero. Hence, by definition, a_i and a_j are structurally orthogonal.

To prove the 'only if' part of the statement, assume that columns a_i and a_j are structurally orthogonal. Then, by definition, there is no $k \in [1, m]$ such that $a_{ki} \neq 0$ and $a_{kj} \neq 0$. This implies that there is no path a_i, r_k, a_j in $G_b(A)$, for any $1 \leq k \leq m$. Hence, by definition, a_i and a_j are non-distance-2 neighbors. □

By Lemma 3.4, finding a consistent partition of the columns of a matrix A is equivalent to finding a *partial* distance-2 coloring of $G_b(A) = (V_1, V_2, E)$ on V_2. (Since V_1 is not involved, the coloring is called partial.) The following result formalizes the equivalence.

Theorem 3.5 *Let A be a nonsymmetric matrix and $G_b(A) = (V_1, V_2, E)$ be its bipartite graph representation. A mapping ϕ is a partial distance-2 coloring of $G_b(A)$ on V_2 if and only if ϕ induces a consistent partition of the columns of A.*

In view of Theorem 3.5, Problem MPP1 is equivalent to the following graph coloring problem (GCP).

Problem 3.6 (*GCP1*) *Given the bipartite graph $G_b(A) = (V_1, V_2, E)$ representing the sparsity structure of a matrix $A \in R^{m \times n}$, find a partial (2,p)-coloring of $G_b(A)$ on V_2 with the least value of p.*

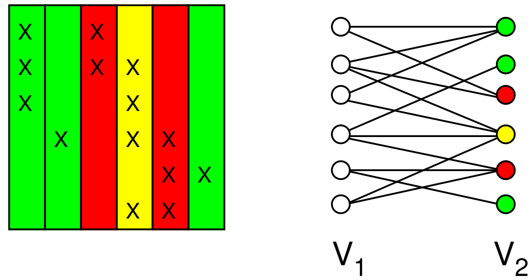


Figure 2: A consistent column partition and its representation as a partial distance-2 coloring

Note that for matrices with a few *dense* rows, a row partition may yield fewer groups than a column partition. Consequently, the matrix problem one needs to solve is MPP1 applied on A^T. In such cases, our graph formulation becomes handy—the equivalent problem is to find a partial distance-2 coloring on the vertex set V_1.

3.3 An Alternative Graph Formulation

Coleman and Moré were the first to formulate MPP1 as a graph coloring problem. In their seminal paper [7], they showed MPP1 to be equivalent to the distance-1 graph coloring problem on the *column intersection graph* of the underlying matrix. In the same work, they developed several coloring heuristics that proved to be quite effective in practice. Implementations of these heuristics were later reported in [5] and made publicly available.

Here, by showing a more general result, we establish the equivalence between the bipartite graph based distance-2 coloring formulation and the column intersection graph based distance-1 coloring formulation of problem MPP1.

The notion of *power* of a graph gives an alternative view to the DkGCP. The kth power of a graph $G = (V, E)$ is the graph $G^k = (V, F)$ where $(u, v) \in F$ if and only if u and v are distance-k neighbors in G. The following equivalence follows immediately.

Lemma 3.7 *Let G^k be the kth power of graph G. A mapping ϕ is a (k, p)-coloring of G if and only if it is a $(1, p)$-coloring of G^k.*

A particular implication of Lemma 3.7 is that distance-2 coloring of a graph is equivalent to distance-1 coloring of the square of the graph. This establishes the equivalence between distance-2 coloring of the bipartite graph and distance-1 coloring of the column intersection graph of a matrix. Specifically, as has been shown in [7], the column intersection graph $G_c(A)$ of a matrix A is isomorphic to the adjacency graph of $A^T A$. We note that $G_c(A)$ is in fact the subgraph of $G_b(A)^2$ induced by the vertices in V_2. For a graph $G = (V, E)$, let the graph induced by $U \subseteq V$ be denoted by $G[U]$.

Lemma 3.8 *Let $G_b(A) = (V_1, V_2, E)$ and $G_c(A) = (V_2, E')$ be the bipartite and column intersection graphs of matrix A. Then, $G_c = {G_b}^2[V_2]$.*

3.4 Comparing The Two Formulations

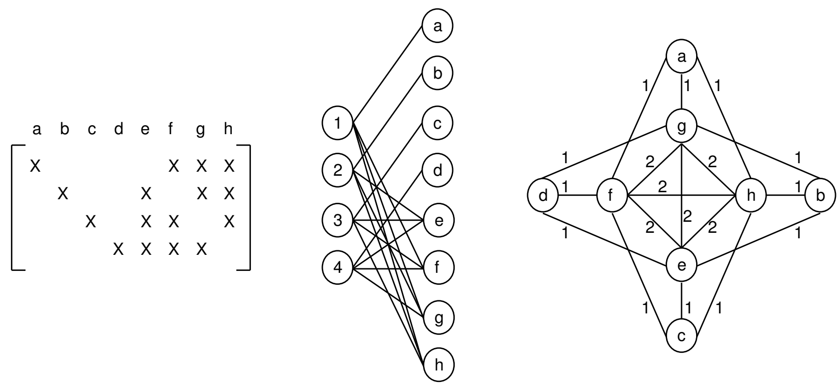


Figure 3: A matrix and its bipartite and intersection graph, resp.

Figure 3 depicts a matrix A, the corresponding bipartite graph $G_b(A) = (V_1, V_2, E)$, and the column intersection graph $G_c(A) = (V_2, E')$.

Note that we have augmented the graph $G_c(A)$ with *edge weights*—$w(a_i, a_j)$ is the size of the intersection of the sets represented by vertices a_i and a_j. In terms of matrix A, $w(a_i, a_j)$ is the total number of rows where both columns a_i and a_j have nonzero entries. In the bipartite graph $G_b(A)$, it corresponds to the number of common neighbors of vertices a_i and a_j.

In Sections 3.4.1 to 3.4.5, we compare and contrast the two graph formulations in terms of flexibility, storage space requirement, ease of graph construction, suitability for generalization, and use of existing software.

3.4.1 Flexibility

Notice that the column intersection graph is not an equivalent, but rather a ‘compressed’, representation of the structure of the underlying matrix. Clearly, some information is lost in the compression process. In particular, given an edge in G_c between two columns, we cannot determine the row(s) where they share nonzero entries. By contrast, the bipartite graph is an equivalent representation of the structure of the matrix. This provides flexibility. For instance, notice that the bipartite graph can be used in a column-only, row-only, or combined row and column partition. The column intersection graph, on the other hand, is applicable only to a column partition. In general, the advantage of the bipartite graph representation (in the context of estimating nonzero matrix entries) is that the representation is decoupled from the eventual technique to be employed and the matrix entries to be determined.

3.4.2 Storage Space Requirement

Although Lemma 3.8 correlates the bipartite graph of a matrix with its column intersection graph, one cannot immediately deduce that one graph is denser than the other. The density of the respective graph depends on the structure of the matrix. Here, we make a rough analysis to show that for sparse matrices of practical interest, the column intersection graph is likely to be denser than the bipartite counterpart.

Given a matrix A, the graph $G_b(A) = (V_1, V_2, E)$ and the weighted graph $G_c(A) = (V_2, E')$, for a vertex u in G_b, let $N_1(u) = \{v : (u, v) \in E\}$, $d_1(u) = |N_1(u)|$, and let the average degree-1 in the sets V_1 and V_2 of G_b be $\overline{\delta}_1(V_1)$ and $\overline{\delta}_1(V_2)$, respectively. Further, let $\overline{w}$ denote the *average edge weight* in G_c. Then,

$$\begin{aligned}
\sum_{e \in E'} w(e) &= \frac{1}{2} \cdot \sum_{u \in V_2} \sum_{v \in N_1(u)} (d_1(v) - 1) \\
|E'| \cdot \overline{w} &\approx \frac{1}{2} \cdot |V_2| \cdot \overline{\delta}_1(V_2) \cdot (\overline{\delta}_1(V_1) - 1) \\
&= \frac{1}{2} \cdot |E| \cdot (\overline{\delta}_1(V_1) - 1) \\
|E'| &= |E| \cdot (\frac{\overline{\delta}_1(V_1) - 1}{2\overline{w}})
\end{aligned}$$

Therefore, as long as $\frac{\overline{\delta}_1(V_1) - 1}{2\overline{w}} > 1$, the column intersection graph is likely to have more edges (and hence requires more storage space) than the bipartite graph of the matrix.

3.4.3 Ease of Construction

The sparsity structure of matrix A directly, without any further computation, gives the corresponding bipartite graph $G_b(A)$. In principle, the data structure used to represent A can be used for implementing algorithms that use $G_b(A)$. By contrast, $G_c(A)$ has to be computed. As the following Lemma states, the time required for the computation of $G_c(A)$ is proportional to the number of edges in $G_c(A)$.

Lemma 3.9 *Given a graph $G_b(A) = (V_1, V_2, E)$, the time required for constructing $G_c(A)$ is $T_{const} = O(|V_2|(\overline{\delta}_1(V_2)(\overline{\delta}_1(V_1) - 1))$.*

It should however be noted that once the graph $G_c(A)$ is computed, a subsequent distance-1 coloring of $G_c(A)$ can be done faster than a distance-2 coloring of $G_b(A)$. As we shall show in Section 3.6, the overall time required for constructing and distance-1 coloring of $G_c(A)$ is of the same order as the time required for a direct distance-2 coloring of $G_b(A)$.

3.4.4 Unification

As has been mentioned earlier, there exist a dozen variants of coloring problems in the context of derivative matrix estimation depending on the type of matrix to be evaluated and the method used for estimation. One of the key advantages of a bipartite graph formulation for Jacobian estimation in this regard is that it led to the discovery of a prototypical problem – the distance-2 coloring problem. As we shall show in the coming sections, a coloring problem in each scenario of derivative matrix estimation turns out to be some relaxed variant of distance-2 coloring. The identification of a prototypical (generic) problem is particularly useful in developing new algorithms for the specialized variants.

3.4.5 Use of Existing Software

Serial program packages that implement various practically effective distance-1 coloring heuristics exist [5, 6]. For matrix partitioning problems where a column intersection graph based formulation can be applied, these packages can be readily used. On the other hand, since distance-2 coloring is a prototypical model in our context, efficient programs, including parallel ones, for the distance-2 coloring problem need to be developed.

3.5 Hypergraph Formulation

In this subsection, we introduce yet another perspective from which MPP1 could be looked at. This view makes use of hypergraphs, a generalization of graphs. Our major goal in introducing the hypergraph formulations is to provide as complete theory as possible. For example, we use the hypergraph formulations to show (from a different perspective) the equivalence between the two graph formulations discussed earlier.

Let us begin by defining a few concepts that we need for our formulation. A *hypergraph* $H = (V, E)$ consists of a finite set V of *vertices* and a collection E of non-empty subsets (of any cardinality) of V called *edges*. Note that the edges in a given hypergraph could be of different size (cardinality). A hypergraph is called r*-uniform* if all of its edges are of size r. Thus, a graph is a 2-uniform hypergraph.

We denote the vertex set and edge set of a hypergraph H by $V(H)$ and $E(H)$, respectively. When the hypergraph under consideration is clear from the context, we may use just V and E to refer to these sets.

The line graph $L(H)$ of hypergraph H is a graph where the vertices of $L(H)$ are the edges of H and the edges of $L(H)$ are pairs of intersecting edges of H.

Figure 4 (left) is a simple example that shows a hypergraph consisting of five vertices and three edges, two edges of size three and one edge of size two. The picture on the right shows the line graph of the hypergraph on the left.

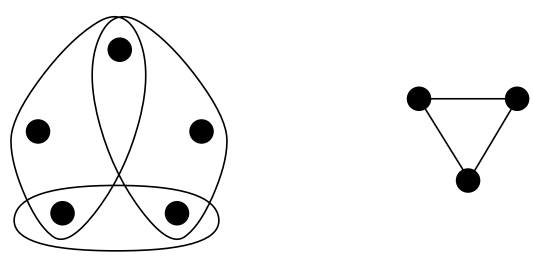


Figure 4: A hypergraph and its line graph

Given a hypergraph $H = (V, E)$ a *strong vertex coloring* (SVC) of H is a mapping $\phi : V \to \{1, 2, \ldots, p\}$ such that for every edge $e \in E$ and every pair of vertices $\{u, v\} \subseteq e$, $\phi(u) \neq \phi(v)$. In other words, in a strong vertex coloring of a hypergraph H, we require that the vertices of every edge of H be assigned distinct colors. The more common variant of hypergraph coloring, called *weak vertex coloring*, requires that no edge of the hypergraph be monochromatic; i.e., at least two different colors are used by the vertices of every edge of the hypergraph.

An *edge coloring* of a hypergraph $H = (V, E)$ is a mapping $\phi : E \to \{1, 2, \ldots, p\}$ such that for edges e_1 and e_2 in E, $\phi(e_1) \neq \phi(e_2)$ whenever $e_1 \cap e_2 \neq \emptyset$.

The strong (weak) vertex coloring problem for hypergraphs asks for (an appropriate) vertex coloring with the least number of colors. Similarly, the edge coloring problem for hypergraphs asks for an edge coloring with the least number of colors.

Some (theoretical) aspects of strong and weak vertex coloring for hypergraphs are studied, for instance, in []. The terms strong and weak coloring were used in []. We have not come across any previous work where hypergraph edge coloring is used.

To achieve our goal of formulating MPP1 using hypergraphs, we introduce two new ways in which the sparsity structure of a matrix can be represented.

Let A be an $m \times n$ 0-1 matrix (representing the sparsity structure of some Jacobian). Define the *column-oriented* hypergraph representation of A as the hypergraph $H_c(A)$ where $V(H_c(A)) = \{v_1, v_2, \ldots, v_n\}$ and each v_j corresponds to the jth column of A, and $E(H_c(A)) = \{e_1, e_2, \ldots, e_m\}$ and each $e_i = \{v_j : a_{ij} = 1\}$ 'corresponds'[1] to the ith row of A. Similarly, define the *row-oriented* hypergraph representation of A as the hypergraph $H_r(A)$ where $V(H_r(A)) = \{v'_1, v'_2, \ldots, v'_m\}$ and each v'_i corresponds to the ith row of A, and $E(H_r(A)) = \{e'_1, e'_2, \ldots, e'_n\}$ and each $e'_j = \{v'_i : a_{ij} = 1\}$ 'corresponds' to the jth column of A.

Observe that hypergraphs $H_c(A)$ and $H_r(A)$, being two representations of the same object, are closely related. Let the *affiliation* of vertex v in hypergraph H, denoted by $\alpha_H(v)$, be the set of edges that contain v. Let two sets be called *index-wise equivalent* if they are of the same cardinality and their correponding elements have the same indices. For example, sets $S = \{s_1, s_5, s_8\}$ and $R = \{r_1, r_5, r_8\}$ are index-wise equivalent. One can then state the following relationships between $H_c(A)$ and $H_r(A)$.

- $|V(H_c(A))| = |E(H_r(A))| = n$.
- $|E(H_c(A))| = |V(H_r(A))| = m$.
- $\alpha_{H_c(A)}(v_i)$ and $e'_i \in E(H_r(A))$ are index-wise equivalent.
- $\alpha_{H_r(A)}(v'_i)$ and $e_i \in E(H_c(A))$ are index-wise equivalent.

Using these facts we state a result that summarizes equivalence relationships that underlie the graph and hypergraph theoretic formulations of MPP1.

Theorem 3.10 *Let A be an $m \times n$ 0-1 matrix. Let $H_c(A)$ and $H_r(A)$ be the column-oriented and row-oriented hypergraph representations of A, respectively, and let $G_b(A)$ be the bipartite graph representation of A. Let ϕ be a*

[1]The quotation marks are used to reflect that a row is perceived as a set of columns at which it has nonzero entries.

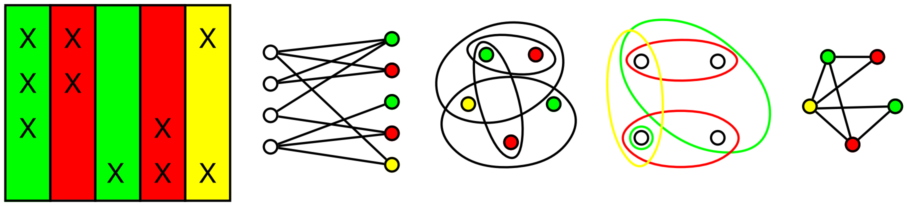


Figure 5: Hypergraph colorings

mapping $S \rightarrow \{1, 2, \ldots, p\}$ where S is a finite set and p is a positive integer. Then, the following statements are equivalent.

i. *ϕ is a strong vertex coloring of hypergraph $H_c(A)$.*

ii. *ϕ is an edge coloring of hypergraph $H_r(A)$.*

iii. *ϕ is a vertex coloring of the line graph $L(H_r(A))$ of hypergraph $H_r(A)$.*

iv. *ϕ is a partial distance-2 coloring of bipartite graph $G_b(A)$.*

v. *ϕ induces a consistent column partition of matrix A.*

Proof:

(i)$\Leftrightarrow$ (v)

Assume that ϕ is a SVC of $H_c(A)$. Clearly ϕ induces a partition of the columns of A. It remains to show that the partition is indeed consistent. Suppose it is not. This means that there exists a group in the partition that contains a pair of columns both of which have nonzeros at some common row position. This in turn means that the vertices in $V(H_c(A))$ corresponding to these columns have been assigned the same color even though they are contained in an edge in $E(H_c(A))$, contradicting the assumption that ϕ is a SVC. Assume now that $\{C_1, \ldots, C_p\}$ is a consistent partition of the columns of A. Construct a vertex coloring ϕ of $H_c(A)$ in such a way that vertices corresponding to columns in C_i are assigned color i. We want to show that ϕ is a SVC of $H_c(A)$. Suppose it is not. Then there must exist an edge in $H_c(A)$ where two of its vertices are assigned the same color. This happens only if the corresponding two columns belonged to the same group in the partition, contradicting the assumption that the partition is consistent.

(i) $\Leftrightarrow$ (ii)

Assume that ϕ is a SVC of $H_c(A)$. Construct an edge coloring ϕ' of $H_r(A)$ as follows: assign each edge in $E(H_r(A))$ the same color as that assigned to the corresponding vertex in $V(H_c(A))$. We claim that ϕ' is in fact an edge coloring of $H_r(A)$. Suppose it is not. This could happen only if there exists some pair of edges e'_x and e'_y in $E(H_r(A))$ such that $e'_x \cap e'_y \neq \emptyset$ and $\phi'(e'_x) = \phi'(e'_y)$. This in turn means that there exists some edge $e \in E(H_c(A)$ that contains vertices v_x and v_y (corresponding to edges e'_x and e'_y) where $\phi(v_x) = \phi(v_y)$, contradicting the assumption that ϕ is a SVC of $H_c(A)$. The other direction can be shown in a similar manner.

(ii)$\Leftrightarrow$ (iii)

Follows directly from the definition of a line graph of a hypergraph.

(ii)$\Leftrightarrow$ (iv)

Consider the vertices in an edge $e \in E(H_c(A))$. Note that these vertices are exactly two units apart from each other in $G_b(A)$. □

Remark 3.11 *The line graph $L(H_r(A))$ of hypergraph $H_r(A)$ is nothing but the column intersection graph $G_c(A)$ of matrix A.*

Using Remark 3.11, it is to be recalled that the equivalence among statements (iii) through (iv) of Theorem 3.10 has actually been established earlier in this section.

In light of Theorem 3.10, MPP1 on A is equivalent to the SVC problem on hypergraph $H_c(A)$ and to the edge coloring problem on hypergraph $H_r(A)$.

As stated earlier, the hypergraph coloring formulations are introduced for the sake of completeness and will not be pursued any further in this paper.

3.6 Algorithms

For any fixed integer $k \geq 1$, the distance-k graph coloring problem is NP-hard [26]. Thus, in practice, one is bound to rely on using *approximation algorithms* or heuristics. An algorithm $\mathcal{A}$ is said to be a γ-approximation algorithm for a minimization problem if its runtime is polynomial in the input size and if for every problem instance $\mathcal{I}$ with an optimal solution $OPT(\mathcal{I})$, the solution $\mathcal{A}(\mathcal{I})$ output by $\mathcal{A}$ is such that $\frac{\mathcal{A}(\mathcal{I})}{OPT(\mathcal{I})} \leq \gamma$. The *approximation ratio* $\gamma \geq 1$, and the goal is to make γ as close to unity as possible. If no such guarantee can be given for the quality of an approximate solution obtained by a polynomial time algorithm, the algorithm is usually referred to as a heuristic.

In the case of distance-1 coloring, there exist several good, practically effective heuristics [7]. In this section we show that some of the ideas used in the distance-1 coloring heuristics can be adapted to the distance-2 coloring case by extending the notion of *neighborhood*. The algorithms we present in this paper are *greedy* in nature, i.e., the vertices of a graph are processed in some order and at each step a decision that looks best at the moment (and that will not be reversed later) is made.

In Section 3.6.1, we present a generic greedy distance-2 coloring algorithm and give a detailed analysis of its performance both in terms of computation time and number of colors used. In later sections, adaptations of this algorithm, tailored to the various coloring problems of our concern will be presented.

3.6.1 A Distance-2 Coloring Algorithm

A simple approach for an approximate distance-2 coloring of a graph $G = (V, E)$ is to visit the vertices in some order, each time assigning a vertex the smallest color that is not used by any of its distance-2 neighbors. See Figure 6 for an example of a proper distance-2 coloring of a graph.

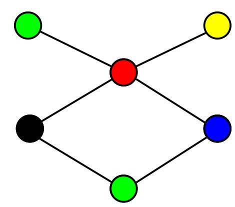


Figure 6: A distance-2 coloring of a graph

Note that the degree-2 of a vertex u in G is bounded by Δ^2, i.e., $d_2(u) \leq \sum_{w \in N_1(u)} d_1(w) \leq \Delta \cdot d_1(u) \leq \Delta^2$. Thus, since the vertices in G can always be distance-2 colored trivially using $|V|$ different colors, it is always possible to color a vertex using a value from the set $\{1, 2, \ldots, min\{\Delta^2 + 1, |V|\}\}$. Algorithm GreedyD2Coloring, outlined below, uses this as it colors the vertices of the graph in an arbitrary order. In the algorithm, color(v) is the color assigned to vertex v and forbiddenColors is a vector of size $C_{max} = \min\{\Delta^2 + 1, |V|\}$ used to mark the colors that cannot be assigned to a particular vertex. Specifically, forbiddenColors$(c) = v$ indicates that color c cannot be assigned to vertex v.

Lemma 3.12 *GreedyD2Coloring finds a distance-2 coloring in time* $O(|V|\overline{\delta}_2)$.

Proof: We first show correctness. In step i of the algorithm, the color used by each of the distance-2 neighbors of vertex v_i is marked (using v_i) in the vector forbiddenColors. Thus, at the end of the inner for loop, the set of colors that are allowed for vertex v_i is the set of indices in forbiddenColors where

```
GreedyD2Coloring(G = (V, E))

for each v_i ∈ V do
    for each colored vertex u ∈ N_2(v_i) do
        forbiddenColors(color(u)) = v_i
    end-for
    color(v_i) = min{c : forbiddenColors(c) ≠ v_i}
end-for
```

the mark used is different from v_i. The minimum value in this set is thus the smallest allowable color for vertex v_i. Notice that the vector forbiddenColors does not need to be initialized at every step as the marker v_i is used *only* in step i.

Turning to complexity, note that marking the forbidden colors at step i of the algorithm takes $O(d_2(v_i))$ time. Finding the smallest allowable color to v_i can be done within the same order of time by scanning forbiddenColors sequentially until the first index c where a value other than v_i is stored is found. The total time is thus proportional to $\sum_{v \in V} d_2(v) = O(|V|\overline{\delta}_2)$. □

We now analyze the quality of the solution provided by GreedyD2Coloring.

Observe that the distance-1 neighbors of a vertex in a graph G form a *clique* in the square of the graph. A clique is a set of vertices in which the vertices are mutually adjacent to each other. This observation immediately provides a lower bound on $\chi_2(G)$, the 2-chromatic number of G.

Lemma 3.13 *For any graph G, $\chi_2(G) \geq \Delta + 1$.*

Proof: Observe that there exists a clique of size $\Delta + 1$ in the square graph G^2. □

Note that G^2 may contain a clique of size larger than $\Delta + 1$. In general, the maximum clique size in G^2 is the tightest lower bound on $\chi_2(G)$.

Let the number of colors used by GreedyD2Coloring on a graph $G = (V, E)$ be $\chi_2{}^{greedy}(G)$. Then using the lower bound given in Lemma 3.13, we get the following theorem and its corollary.

Theorem 3.14 $\Delta + 1 \leq \chi_2(G) \leq \chi_2{}^{greedy}(G) \leq min\{\Delta^2 + 1, |V|\}$.

Corollary 3.15 *GreedyD2Coloring is an $O(\sqrt{|V|})$-approximation algorithm.*

Proof: The approximation ratio γ is at most $\frac{1}{\Delta+1} \cdot min\{\Delta^2+1, |V|\}$. There are two possibilities to consider. In the first case $\Delta^2+1 < |V|$. This implies $\Delta = O(\sqrt{|V|})$ and $\gamma = \frac{\Delta^2+1}{\Delta+1} = O(\Delta) = O(\sqrt{|V|})$. In the second case $|V| < \Delta^2+1$. This implies $\Delta = \Omega(\sqrt{|V|})$ and $\gamma = \frac{|V|}{\Delta+1} = O(\sqrt{|V|})$. □

Note that for practical problems, such as problems that arise in solving PDEs using good finite element discretizations, $\Delta^2 + 1 \ll |V|$, making GreedyD2Coloring an $O(\Delta)$-approximation algorithm.

The actual number of colors used by GreedyD2Coloring depends on the order in which the vertices are visited. In GreedyD2Coloring, an arbitrary ordering is assumed. A solution with fewer number of colors can be expected if a more elaborate ordering criterion is used. For example, the ideas in *largest degree first* and *incidence degree ordering* for distance-1 coloring [7] can be adapted to the distance-2 coloring case.

As a final remark on the complexity of GreedyD2Coloring, we show that the algorithm runs in linear time in the number of vertices for certain sparse graphs. Let $\delta_2 = \frac{1}{|V|}\sum_{u\in V} d_1(u)^2$ and let the *standard deviation* of degree-1 in $G = (V, E)$ be given by

$$\sigma^2 = \frac{1}{|V|}\sum_{v\in V} (d_1(v) - \overline{\delta}_1)^2.$$

Then,

$$\begin{aligned}
|V|\sigma^2 &= \sum_{v\in V} d_1(v)^2 + \sum_{v\in V} \overline{\delta}_1^{\,2} - 2\sum_{v\in V} d_1(v)\overline{\delta}_1 \\
&= |V|\delta_2 + |V|\overline{\delta}_1^{\,2} - 2\overline{\delta}_1 \sum_{v\in V} d_1(v) \\
&= |V|\delta_2 + |V|\overline{\delta}_1^{\,2} - 2|V|\overline{\delta}_1^{\,2} \\
&= |V|\delta_2 - |V|\overline{\delta}_1^{\,2}
\end{aligned}$$

Rewriting we get,

$$\delta_2 = \overline{\delta}_1^{\,2} + \sigma^2.$$

Noting that $\delta_2 \geq \overline{\delta}_2$, we get the following corollary to Lemma 3.12.

Corollary 3.16 *GreedyD2Coloring has time complexity* $O(|V|(\overline{\delta}_1^{\,2} + \sigma^2))$.

Since $\overline{\delta}_1 = \frac{2|E|}{|V|}$, the complexity expression in Corollary 3.16 reduces to $O(\frac{|E|^2}{|V|})$ for graphs where $\sigma < \overline{\delta}_1$. In particular, for sparse graphs, where $|E| = O(|V|)$, the time complexity of GreedyD2Coloring becomes $O(|V|)$.

3.6.2 A Partial Distance-2 Coloring Algorithm

Here we modify GreedyD2Coloring slightly to make it suitable for solving the partial distance-2 coloring problem GCP1 (our graph formulation of MPP1). The resulting algorithm, called GreedyPartialD2Coloring, is outlined in the sequel.

Let $G_b = (V_1, V_2, E)$ be the input to our algorithm. For any vertex $v \in V_2$, the number of vertices at distance *exactly* two units from v is at most $\Delta(V_2)(\Delta(V_1) - 1)$. Thus, vertex v can always be assigned a color from the set $\{1, 2, \ldots, C_{max}\}$, where $C_{max} = min\{\Delta(V_2)(\Delta(V_1) - 1)) + 1, |V_2|\}$. In Algorithm GreedyPartialD2Coloring the vector forbiddenColors is of size C_{max}.

GreedyPartialD2Coloring($G_b = (V_1, V_2, E)$)

for each $v \in V_2$ **do**
    for each $u \in N_1(v)$ **do**
        for each colored vertex $w \in N_1(u)$ **do**
            forbiddenColors(color(w)) $= v$
        end-for
    end-for
    color(v) $= \min\{c :$ forbiddenColors$(c) \neq v\}$
end-for

Lemma 3.17 gives the time complexity of the algorithm.

Lemma 3.17 *GreedyPartialD2Coloring has time complexity* $O(|V_2|\overline{\delta}_1(V_2)(\overline{\delta}_1(V_1) - 1))$.

Note that one can state the following corollary to Lemma 3.13. Let $\chi_{2p}(G_b)$ denote the minimum number of colors required for a partial distance-2 coloring of $G_b = (V_1, V_2, E)$ on V_2.

Corollary 3.18 $\chi_{2p}(G_b) \geq \Delta(V_1)$.

It is interesting to see the meaning of Corollary 3.18 in terms of problem MPP1. The result says that $\Delta(V_1)$, which is equal to the maximum number of nonzeros in a row of the underlying matrix, is a lower bound on the least number of groups required in a consistent column partition. This lower bound was also observed by Coleman and Moré [7]. It should however be pointed out that a better lower bound can be envisioned for some matrix structures. More specifically, the maximum clique size in ${G_b}^2[V_2]$ is the tightest lower bound.

3.6.3 A Distance-1 Coloring Algorithm

As stated earlier, a distance-1 coloring formulation for MPP1 was provided in [7] using the column intersection graph. Here, to serve us as basis for later comparison, we describe Algorithm GreedyD1Coloring, which is the distance-1 analog of GreedyD2Coloring. In GreedyD1Coloring the vector forbiddenColors is of size $\Delta + 1$ where Δ is the maximum degree-1 in the input graph to the algorithm.

GreedyD1Coloring($G = (V, E)$)

for each $v_i \in V$ **do**
    for each colored vertex $u \in N_1(v_i)$ **do**
        forbiddenColors(color(u)) $= v_i$
    end-for
    color(v_i) $= \min\{c :$ forbiddenColors(c) $\neq v_i\}$
end-for

The following result is straightforward.

Lemma 3.19 *GreedyD1Coloring finds a distance-1 coloring in time* $O(|V|\overline{\delta}_1) = O(|E|)$.

From Lemmas 3.9, 3.17 and 3.19 it follows that the time required for the construction of the column intersection graph plus the computation of a (greedy) distance-1 coloring is asymptotically the same as the time required for the direct computation of a (greedy) partial distance-2 coloring. This means that the two formulations are (asymptotically) comparable in terms of overall computation time. In practice, however, the runtimes while using the two approaches differ considerably. Our experimental results reported in the next subsection demonstrate this fact.

3.7 Experimental Results

We made an experimental comparison between the bipartite graph based distance-2 coloring formulation and the column intersection graph based distance-1 coloring formulation of the Jacobian estimation problem. The comparison focuses on quality of coloring obtained, overall execution time, and storage space requirement.

The algorithms in our test (including those reported in Section 4.4) are implemented in C, and the experiments are conducted on an Intel Pentium 4, 2.53 GHz machine with 1 GB memory and 512KB cache, running Linux 2.4.20/RedHat 8.0.

Matrix	m	n	nnz	$\eta(\%)$	κ_{max}	κ_{min}	κ_{avg}	ρ_{max}	ρ_{min}	ρ_{avg}
cre_a	3516	7248	18168	0.07	14	1	2.51	360	0	5.17
dfl001	6071	12230	35632	0.05	14	1	2.91	228	2	5.87
ken_11	14694	21349	49058	0.02	3	1	2.30	122	1	3.34
stocfor3	16675	23541	76473	0.02	18	1	3.25	15	1	4.59
ken_13	28632	42659	97246	0.008	3	1	2.28	170	1	3.40
pds_10	16558	49932	107605	0.01	3	1	2.16	96	1	6.50
maros_r7	3136	9408	144848	0.50	46	1	15.40	48	5	46.19
lhr10	10672	10672	232633	0.20	36	1	21.80	63	1	21.80
pds_20	33874	108175	232647	0.006	3	1	2.15	96	0	6.87
cre_d	8926	73948	246614	0.04	13	1	3.33	808	0	27.63
cre_b	9648	77137	260785	0.04	14	1	3.38	844	0	27.03
e30r2000	9661	9661	306356	0.33	62	8	31.71	62	8	31.71
lhr14	14270	14270	307858	0.15	36	1	21.57	63	1	21.57
ken_18	105127	154699	358171	0.002	3	1	2.31	325	1	3.40
af23560	23560	23560	484256	0.09	21	10	20.55	21	11	20.55
e40r0100	17281	17281	553956	0.19	62	8	32.05	62	8	32.05
cage11	39082	39082	559722	0.04	31	3	14.32	31	3	14.32
lhr34	35152	35152	764014	0.06	36	1	21.73	63	1	21.73
lhr71c	70354	70304	1528092	0.24	36	1	21.73	63	1	21.73
cage12	130228	130228	2032536	0.01	33	5	15.61	33	5	15.61
fit2d	25	10524	129042	49.00	17	1	12.26	10500	1427	5161.68
osa_07	1118	25067	144812	0.52	6	1	5.78	17613	18	129.53

Table 1: Matrix statistics

3.7.1 Test Matrices

Our testbed consists mainly of matrices obtained from the University of Florida Sparse Matrix Collection [13].

Table 1 lists some relevant structural statistics about the test matrices. The number of rows, columns, and nonzeros in each matrix is listed under columns m, n, and nnz, respectively. The density of a matrix in percent is given in column η ($= \frac{nnz}{m \cdot n} \times 100$). The maximum, minimum, and average number of nonzeros per *column* of a matrix are listed under κ with an appropriate subscript. The corresponding figures per *row* are given under the various ρ's.

3.7.2 Results

Table 2 shows the results of Algorithm GreedyPartialD2Coloring run on the bipartite graph representation of the test matrices. The left-half of the table lists information on the underlying graph. Under column $|G|$ is given the sum of the number of vertices ($|V|$) and edges ($|E|$) of each graph. The quantity $|G|$ is used as a basis for assessing storage space requirement. The maximum, minimum, and average degree-1 in the graph are given under Δ, δ and $\overline{\delta}$, respectively. Note the relationship between the graph and matrix structural statistics given in Tables 2 and 1: $|V| = m + n$, $|E| = nnz$,

Matrix	$\|V\|$	$\|E\|$	$\|G\|$	Δ	δ	$\bar{\delta}$	K (ρ_{max})	T_{G_b}	T_{col}	T_{tot}
cre_a	10764	18168	28932	360	0	3.38	360 (360)	0	0.01	0.01
dfl001	18301	35632	53933	228	1	3.89	228 (228)	0	0.01	0.01
ken_11	36043	49058	85101	122	1	2.72	130 (122)	1	0.00	1.00
stocfor3	40216	76473	116689	18	1	3.80	16 (15)	1	0.03	1.03
ken_13	71291	97246	168537	170	1	2.73	176 (170)	1	0.02	1.02
pds_10	66490	107605	240585	96	1	3.23	96 (96)	1	0.01	1.01
maros_r7	12544	144848	153392	48	1	23.10	74 (48)	1	0.07	1.07
lhr10	21344	232633	253977	63	1	21.80	65 (63)	1	0.10	1.10
pds_20	142049	232647	374696	96	0	3.28	96 (96)	2	0.03	2.03
cre_d	82874	246614	329488	808	0	5.95	813 (808)	2	0.34	2.34
cre_b	86785	260785	304570	844	0	6.01	845 (844)	2	0.34	2.34
e30r2000	19322	306356	325678	62	8	31.71	65 (62)	2	0.07	2.07
lhr14	28540	307858	336398	63	1	21.57	65 (63)	2	0.11	2.11
ken_18	259826	358171	617997	325	1	2.76	330 (325)	7	0.23	7.23
af23560	47120	484256	531376	21	10	20.55	32 (21)	3	0.09	3.09
e40r0100	34562	553956	588518	62	8	32.05	66 (62)	4	0.13	4.13
cage11	78164	559722	637886	31	3	14.32	81 (31)	4	0.12	4.12
lhr34	70304	764014	834318	63	1	21.73	65 (63)	6	0.25	6.25
lhr71c	140608	1528092	1668700	63	1	21.73	65 (63)	12	0.45	12.45
cage12	260456	2032536	2292992	33	5	15.61	96 (33)	15	0.37	15.37
Total1			9947763				3764 (3573)	67	2.78	69.78
fit2d	10549	129042	139591	10500	1	24.47	10501 (10500)	0	4.63	4.63
osa_07	26185	144812	170997	17613	1	11.06	17613 (17613)	0	5.07	5.07
Total2			310588				28114 (28113)	0	9.70	9.70

Table 2: Results of partial $d2$-coloring on bipartite graph $G_b(A)$

$\Delta = max\{\kappa_{max}, \rho_{max}\}$, and $\delta = min\{\kappa_{min}, \rho_{min}\}$.

The right-half of Table 2 lists coloring and timing information. Column K lists the number of colors used by GreedyPartialD2Coloring and the number in parenthesis in the same column shows the lower bound ρ_{max} on the optimal number of colors. The time used (in seconds) for constructing the bipartite graph is given in column T_{G_b}, and the time spent on coloring is listed under T_{col}. The last column gives the sum of the previous two.

Table 3 lists results obtained when GreedyD1Coloring is run on the column intersection graph of each test matrix. The column intersection graph is obtained by first constructing the bipartite graph and then computing the subgraph of the square graph induced by the vertex set corresponding to the columns of the underlying matrix. See Lemma 3.8 for this relationship between the bipartite and column intersection graphs. In our experiments, we used the routine ConstructG_c shown in Figure 7 to compute an intersection graph from a bipartite graph. In this routine, the vector marked of size $|V_2|$ is used to avoid multiple edges between two vertices in the intersection graph that are connected by several paths of length two in the bipartite graph.

In Table 3 the time spent on the respective graph constructions is given under columns T_{G_b} and T_{G_c}. The other columns are defined in a similar manner as in Table 2.

Matrix	$\|V\|$	$\|E\|$	$\|G\|$	Δ	δ	$\overline{\delta}$	T_{G_b}	T_{G_c}	T_{col}	T_{tot}
cre_a	7248	253411	260659	454	1	69.93	0	0	0.01	0.01
dfl001	12230	250976	263206	423	2	41.04	0	4	0.01	4.01
ken_11	21349	459921	481270	138	1	43.09	0	2	0.01	2.01
stocfor3	23541	125969	149510	39	1	10.70	0	0	0.01	0.01
ken_13	42659	1158664	1201323	186	2	54.32	1	9	0.01	10.01
pds_10	49932	594681	644613	106	1	23.82	1	8	0.01	9.01
maros_r7	9408	610760	620168	314	4	129.84	1	8	0.01	9.01
lhr10	10672	431411	442083	101	1	80.85	1	5	0.02	6.02
pds_20	108175	1325891	1434066	115	1	24.51	2	10	0.02	12.02
cre_d	73948	21347885	21421833	1124	2	577.38	2	76	0.25	78.25
cre_b	77137	20852569	20929706	1168	2	540.67	2	268	0.75	270.75
e30r2000	9661	688848	698509	209	42	142.61	2	12	0.02	14.02
lhr14	14270	572463	586733	101	1	80.23	1	10	0.00	11.00
ken_18	154699	8412174	8566873	340	1	108.76	8	43	0.15	51.15
af23560	23560	1210004	1233564	107	41	102.72	2	12	0.02	14.02
e40r0100	17281	1254328	1288890	209	42	145.17	3	14	0.01	17.01
cage11	39082	1887384	1926466	340	7	96.59	4	16	0.02	20.02
lhr34	35152	1417888	1453040	101	1	80.67	5	22	0.03	27.03
lhr71c	70304	2835968	2906272	101	1	80.67	8	39	0.04	47.04
cage12	130228	7550823	7681051	400	12	115.96	15	57	0.11	72.11
Total1			74189835				58	615	1.51	674.51
fit2d	>	100	million	edges						
osa_07	>	100	million	edges						

Table 3: Results of $d1$-coloring on column intersection graph $G_c(A)$

3.7.3 Discussion

A comparison between the results in Tables 2 and 3 reveals, among others, the following points.

Quality The two coloring formulations are equivalent in terms of quality. In particular, as long as the vertices are visited in the same order, greedy distance-1 coloring on a column intersection graph uses exactly the same number of colors as partial distance-2 coloring on a bipartite graph. In Table 3, the column listing the number of colors used by the algorithm is omitted since the corresponding numbers are identical to those listed under column K in Table 2. Another observation regarding quality of coloring is that, as can be seen from Table 2, for the matrices in our experiment, the number of colors used by the greedy algorithm is often close to, and in some cases even equal to, the lower bound ρ_{max} on the optimal value.

Space In general, for the matrices used in our experiments, a column intersection graph is bigger in size than a bipartite graph and hence requires more storage space. The difference in graph size becomes particularly large for matrices with relatively 'dense' rows ('high' ρ_{avg}). For instance, the column intersection graph of matrix cre_b is 68 times as big as the corresponding

```
ConstructG_c
Input: G_b = (V_1, V_2, E)
Output: G_c = (V_2, E')

E' = ∅
for each v ∈ V_2 do
    for each u ∈ N_1(v) do
        for each w ∈ N_1(u) do
            if (w ≠ v)
                if (marked(w) ≠ v)
                    E' = E' ∪ (v, w)
                    marked(w) = v
                end-if
            end-if
        end-for
    end-for
end-for
```

Figure 7: Computing intersection graph from bipartite graph.

bipartite graph. In the case of matrices fit2d and osa_07, the column intersection graph was too big to fit in the available memory space (the graph construction process in Figure 7 was stopped when the number of edges $|E'|$ exceeded 100 million). Considering the sum total of the graph sizes of all other matrices from our testbed, the column intersection graph representation would require more than *seven* times as much storage space as bipartite graph representation. See Figure 8 for a comparative plot of graph sizes using the two approaches.

Time In terms of overall computation time, our experimental results show that a distance-2 coloring approach is significantly faster than a method based on distance-1 coloring. See Figure 9 for a comparative plot. Again, the difference in overall execution time becomes very big for matrices with dense rows. As an example, the overall time used for intersection graph construction and distance-1 coloring for matrix cre_b is 115 times the time used for bipartite graph construction followed by partial distance-2 coloring. Note that for the first twenty matrices, on which both approaches could be tested successfully, the total time spent on first constructing the column intersection graph and then distance-1 coloring is about *ten* times the overall time spent on constructing the bipartite graph and then partial distance-

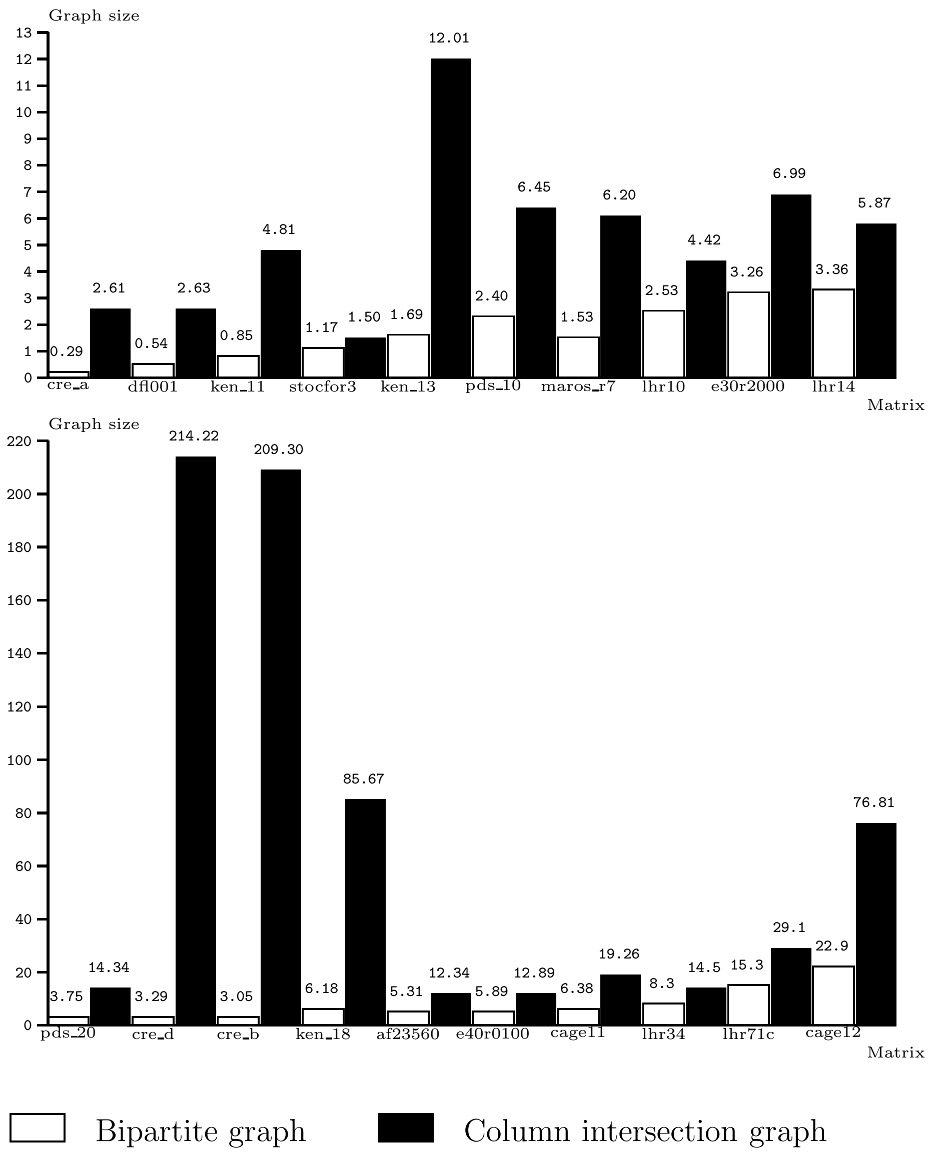
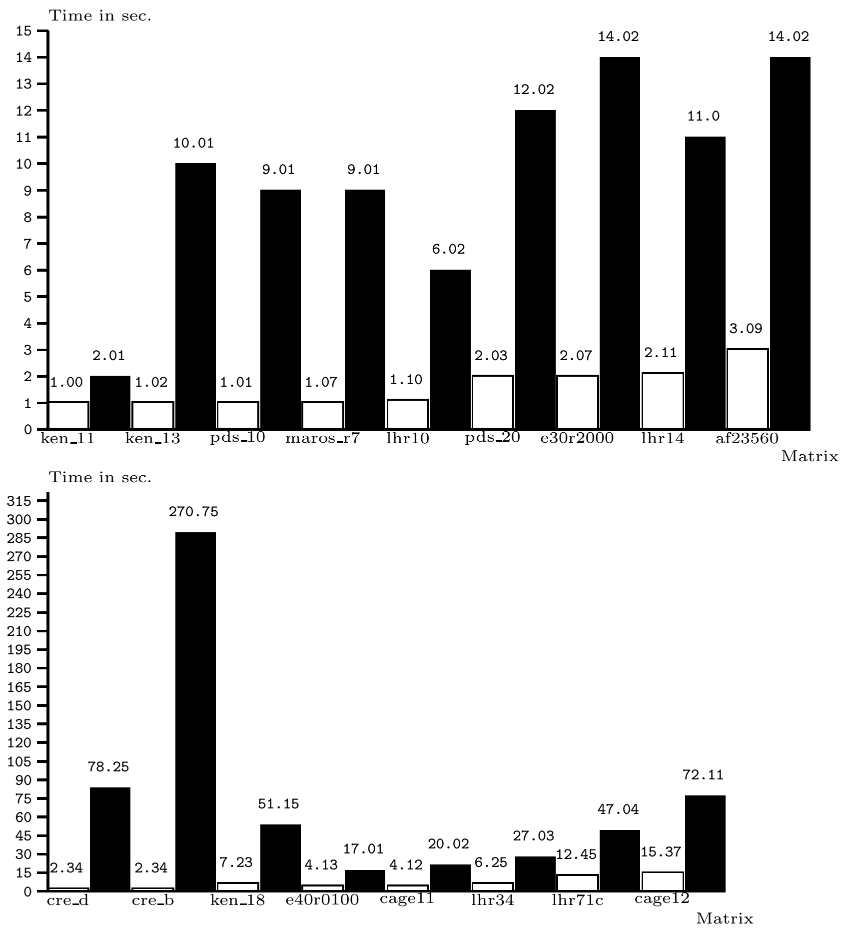


Figure 8: Storage space comparison. Graph size ($|V| + |E|$) in 100,000



Partial $d2$-coloring on bipartite graph $d1$-coloring on column intersection graph

Figure 9: Overall execution (graph construction plus coloring) time comparison.

2 coloring. This rather big difference could be due to the high memory access cost associated with the construction and processing of the column intersection graph.

4 Direct Estimation of Hessians

Given a twice continuously differentiable function $f : R^n \to R$, the **Hessian** of f at the point x is the $n \times n$ symmetric matrix whose (i, j) entry $H(x)_{ij} = \nabla^2 f(x)_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$. When ∇f is available, $\nabla^2 f$ can be approximated by applying the finite difference Formula (1) to the function $F = \nabla f$. We assume that evaluating the gradient $\nabla f(x)$ as a single entity is more desirable than evaluating the components $\partial_1 f(x), \ldots, \partial_n f(x)$ separately.

4.1 The Matrix Partitioning Problem

Let $d_i = \sum_{j \in J' \subseteq \{1,2,\ldots,n\}} e_j$ where e_j is the jth unit vector. Using analogous discussion as in the Jacobian estimation case of the previous section, the problem of interest in the efficient, direct estimation of the Hessian matrix can be stated as follows.

Problem 4.1 *Given the sparsity structure of a symmetric matrix $A \in R^{n\times n}$, find the fewest binary vectors $d_1, d_2, \ldots, d_p$ such that $Ad_1, Ad_2, \ldots, Ad_p$ determine A directly.*

Powell and Toint [33] showed that, in the case of Hessian estimation, in addition to exploiting sparsity using the Curtis, Powell and Reid technique, the number of function evaluations can be reduced further by exploiting symmetry. Coleman and Moré [8] later gave a definition (Definition 4.2) for a more general version of the partition underlying the approach of Powell and Toint.

Definition 4.2 A partition of the columns of a symmetric matrix A is *symmetrically consistent* with a direct determination of A if whenever a_{ij} is a nonzero element of A then either the group containing a_j has no other column with a nonzero in row i, or the group containing a_i has no other column with a nonzero in row j.

See Figure 10 for an example of a symmetrically consistent partition of the columns of a matrix.

Let $\{C_1, C_2, \ldots, C_p\}$ be a symmetrically consistent partition. With each group C_k, associate a binary vector d_k having components $\delta_j = 1$ if a_j belongs to C_k, and $\delta_j = 0$ otherwise. Then,

$$Ad_k = \sum_{\forall a_j} \delta_j a_j = \sum_{a_j \in C_k} a_j.$$

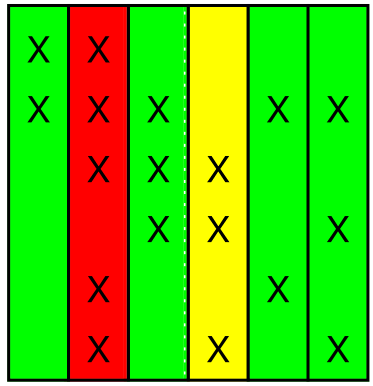


Figure 10: A symmetrically consistent column partition

If $a_{ij} \neq 0$ and column a_j is the only column in C_k with a nonzero in row i, then $a_{ij} = (Ad_k)_i$; alternatively, if a_i is the only column in $C_{k'}$ with a nonzero in row j, then $a_{ji} = (Ad_{k'})_j$. This way all the diagonal entries and half of the off-diagonal nonzero entries of A can be determined with p evaluations of Ad_k. The other half of the off-diagonal nonzero elements are then obtained by symmetry.

If a consistent partition (rather than a symmetrically consistent one) is used to compute a symmetric matrix A, the estimate for a_{ij} may actually be different from that of a_{ji} due to truncation error. Thus, using a symmetrically consistent partition to compute half of the off-diagonal nonzero elements of a matrix and determining the other half by symmetry is preferable both in terms of reducing computational work and ensuring that the computed matrix is indeed symmetric.

Using Definition 4.2, Problem 4.1 can be restated as follows.

Problem 4.3 (*MPP2*) *Given the sparsity structure of a symmetric matrix $A \in R^{n \times n}$, find a symmetrically consistent partition of the columns of A that has the least number of groups.*

4.2 The Graph Formulation

The *adjacency graph*, since it exploits symmetry, is an appropriate graph representation for the sparsity structure of a Hessian. A Hessian matrix A is often positive definite [4]; hence the assumption in the definition of the adjacency graph $G(A)$ that all diagonal entries in A are nonzero is reasonable.

Let us proceed by presenting an adjacency graph based characterization of structural orthogonality in a symmetric matrix. This characterization is due to McCormick [28].

Lemma 4.4 *Let $A \in R^{n \times n}$ be a symmetric matrix with nonzero diagonal elements and let $G(A) = (V, E)$ be its adjacency graph. Two columns in A are structurally orthogonal if and only if the corresponding vertices in $G(A)$ are non-distance-2 neighbors.*

The proof of Lemma 4.4 is very similar to that of Lemma 3.4 and hence omitted here. Notice that, just as structural orthogonality of two columns in a nonsymmetric matrix corresponds to the vertices in the bipartite graph being at distance at least three edges from each other, structural orthogonality of two columns in a symmetric matrix corresponds to the vertices in the adjacency graph being at distance at least three from each other.

By Lemma 4.4, finding a consistent partition of the columns of a symmetric matrix A is equivalent to finding a distance-2 coloring of the adjacency graph $G(A)$. McCormick first observed this equivalence and used it to suggest algorithms for solving the Hessian estimation problem. However, as has been stated earlier, the symmetry present in A can be exploited to further reduce the number of groups (colors) required. We now consider the graph coloring formulation of MPP2, the partitioning problem where symmetry in A is exploited.

Consider a symmetric matrix A with nonzero diagonal elements and let a_{ij}, $i \neq j$, be any nonzero element in A. Further, let a_{ki}, $k \neq i, j$ and a_{jl}, $l \neq i, j, k$ be any other two nonzero elements. By Definition 4.2, in a symmetrically consistent partition of A,

- columns a_i and a_j should belong to two different groups (this is because both a_{ii} and a_{jj} are nonzero), and

- columns a_j and a_k should belong to two different groups, or columns a_i and a_l should belong to two different groups.

Coleman and Moré [8] characterized the aforementioned conditions in terms of a coloring of the associated adjacency graph. Specifically, they introduced the notion *distance-$\frac{3}{2}$ coloring*[2] to capture these conditions.

Definition 4.5 A mapping $\phi : V \to \{1, 2, \ldots, p\}$ is a *distance-$\frac{3}{2}$ p-coloring* of the graph $G = (V, E)$ if ϕ is a distance-1 p-coloring of G and every path of length three uses at least three colors.

The name 'distance-$\frac{3}{2}$ coloring' is chosen to reflect that it is in a sense 'in-between' distance-1 and distance-2 colorings. In particular, a distance-$\frac{3}{2}$ coloring is a relaxed distance-2 and a restricted distance-1 coloring. As

[2] Coleman and Moré used the term *path-coloring*.

an illustration, observe that a distance-1 coloring requires two colors for every path of length one, a distance-2 coloring requires three colors for every path of length two, and a distance-$\frac{3}{2}$ coloring is a distance-1 coloring further restricted to require three colors for every path of length three (see Figure 11). Note that a 4-cycle requires two, three, and four colors in a distance-1, a distance-$\frac{3}{2}$ and a distance-2 coloring, respectively.

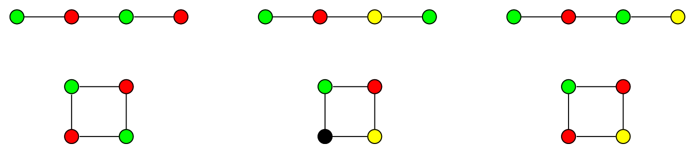


Figure 11: Distance-1, 2, and $\frac{3}{2}$ coloring of P_4 and C_4

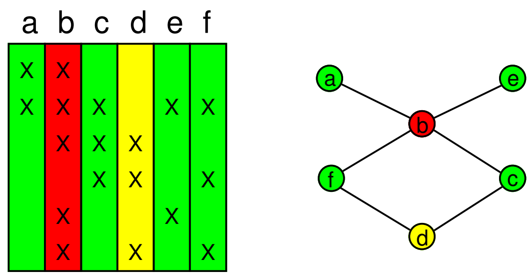


Figure 12: A symmetrically consistent column partition and its representation as a distance-$\frac{3}{2}$ coloring

The following theorem formalizes the connection between symmetrically consistent partition and distance-$\frac{3}{2}$ coloring. The result follows directly from the discussion that led to the definition of distance-$\frac{3}{2}$ coloring.

Theorem 4.6 [Coleman and Moré [8]]
Let A be a symmetric matrix with nonzero diagonal elements and $G(A) = (V, E)$ be its adjacency graph representation. A mapping ϕ is a distance-$\frac{3}{2}$ coloring of $G(A)$ if and only if ϕ induces a symmetrically consistent partition of the columns of A.

By Theorem 4.6, the following problem is equivalent to MPP2.

Problem 4.7 (*GCP2*) *Given the adjacency graph $G(A) = (V, E)$ representing the sparsity structure of a symmetric matrix $A \in R^{n\times n}$ with nonzero diagonal elements, find a $(\frac{3}{2}, p)$-coloring of $G(A)$ with the least value of p.*

4.3 Algorithms

Coleman and Moré [8] showed that the problem of finding a distance-$\frac{3}{2}$ coloring with the minimum number of colors is NP-hard even if the graph is bipartite. Here we discuss two heuristic algorithms for this problem. In finding a valid color to assign a vertex, the first algorithm visits the distance-3 neighbors of the vertex while the second algorithm visits only the distance-2 neighbors. In both algorithms a vector forbiddenColors of size $C_{max} = min\{\Delta^2 + 1, |V|\}$ is used.

4.3.1 The First Distance-$\frac{3}{2}$ Coloring Algorithm

GreedyD$\frac{3}{2}$ColoringAlg1 outlines the first algorithm. The accompanying figure graphically shows the decision made during one of the $|V|$ steps of GreedyD$\frac{3}{2}$ColoringAlg1. The root of the tree corresponds to the vertex v to be colored at the current step. The neighbors of v that are one, two, and three edges away are represented by the nodes at level u, w, and x, respectively. Each tree node corresponds to many vertices of the input graph. A shaded node signifies that the vertex is already colored. The forbidden colors are marked by an f and '?' indicates that whether the color is forbidden or not depends on the color used at level x. The correspondence between the figure and Lines 1, 2 and 3 of the algorithm is obvious.

Notice that in Line 2 of the algorithm, the color of the vertex w in the path (v, u, w) where u is not yet colored is forbidden for vertex v. Later on, when vertex u is colored, the test in Line 1 ensures that u gets a color different from both v and w, making the path use three different colors. Had the requirement in Line 2 not been imposed, a situation in which a path (v, u, w, x) is two-colored could arise. Thus from its construction, the output of GreedyD$\frac{3}{2}$ColoringAlg1 is a valid distance-$\frac{3}{2}$ coloring. The amount of work done in each step of the algorithm is proportional to $d_3(v)$. Thus we get the following result.

Lemma 4.8 *GreedyD$\frac{3}{2}$ColoringAlg1 finds a distance-$\frac{3}{2}$ coloring in time $O(|V|\overline{\delta}_3)$.*

4.3.2 The Second Distance-$\frac{3}{2}$ Coloring Algorithm

The idea behind our second distance-$\frac{3}{2}$ coloring algorithm was first suggested by Powell and Toint [33] in a formulation based on matrices.

Recall that distance-$\frac{3}{2}$ coloring is a relaxed distance-2 coloring. As an illustration, suppose v, u, w, x is a path in a graph. A coloring ϕ in which

GreedyD$\frac{3}{2}$ColoringAlg1$(G = (V, E))$

```
for each v ∈ V do
    for each u ∈ N_1(v) do
        if u is colored                                  (1)
           forbiddenColors(color(u)) = v
        for each colored vertex w ∈ N_1(u) do
            if u is not colored                          (2)
               forbiddenColors(color(w)) = v
            else
               for each colored vertex x ∈ N_1(w), x ≠ u do
                   if (color(x) == color (u))            (3)
                      forbiddenColors(color(w)) = v
                      break
                   end-if
               end-for
            end-if
        end-for
    end-for
    color(v) = min{c : forbiddenColors(c) ≠ v}
end-for
```

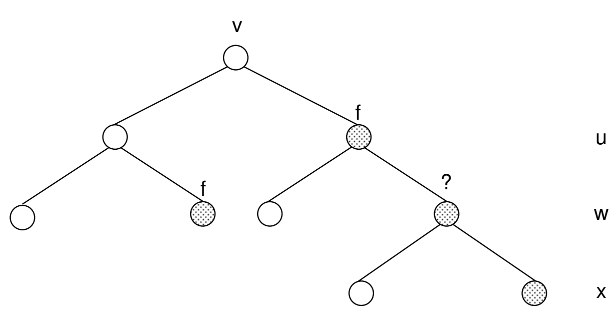


Figure 13: Visualizing a step in GreedyD$\frac{3}{2}$ColoringAlg1

$\phi(v) = \phi(w) = 2$, $\phi(u) = 1$ and $\phi(x) = 3$ is a valid distance-$\frac{3}{2}$ (but not distance-2) coloring on this path.

One way of relaxing the requirement for distance-2 coloring so as to obtain a distance-$\frac{3}{2}$ coloring is to let two vertices at distance of exactly two units from each other share a color as long as the vertex in between them has a color of lower value. More precisely, let v, u, w be a path in G and suppose v and u are colored and we want to determine the color of w. We allow $\phi(w)$ to be equal to $\phi(v)$ as long as $\phi(u) < \phi(v)$. To see that this coloring can always be extended to yield a valid distance-$\frac{3}{2}$ coloring, consider the path v, u, w, x, an extension of path v, u, w in one direction. Now, since $\phi(w) = \phi(v) > \phi(u)$, we cannot let $\phi(x)$ be equal to $\phi(u)$. Obviously, $\phi(x)$ should be different from $\phi(w)$, otherwise it will not be a valid distance-1 coloring. Thus the path v, u, w, x uses three colors, ϕ is a distance-1 coloring and therefore it is a valid distance-$\frac{3}{2}$ coloring. The algorithm that makes use of this idea is given in GreedyD$\frac{3}{2}$ColoringAlg2.

```
GreedyD3/2ColoringAlg2(G = (V, E))

for each v ∈ V do
    for each u ∈ N1(v) do
        if u is colored
          forbiddenColors(color(u)) = v
        for each colored vertex w ∈ N1(u) do
            if u is not colored
              forbiddenColors(color(w)) = v
            else
              if (color(w) < color(u))
                forbiddenColors(color(w)) = v
            end-if
        end-for
    end-for
    color(v) = min{c : forbiddenColors(c) ≠ v}
end-for
```

Clearly, the runtime of GreedyD$\frac{3}{2}$ColoringAlg2 is $O(|V|\overline{\delta}_2)$. Notice, however, that GreedyD$\frac{3}{2}$ColoringAlg1 may use fewer colors than GreedyD$\frac{3}{2}$ColoringAlg2. Figure 15 shows an example where the first algorithm uses three colors in coloring the vertices in their alphabetical order while the second one uses four in doing the same. Figure 14 shows where the two algorithms use the same number of colors.

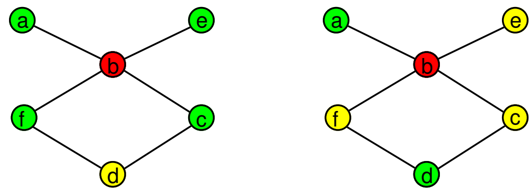


Figure 14: GreedyD$\frac{3}{2}$ColoringAlg1 vs. GreedyD$\frac{3}{2}$ColoringAlg2: (same number of colors)

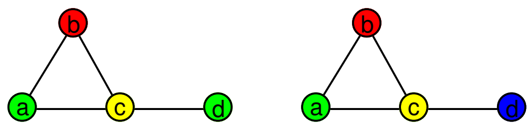


Figure 15: GreedyD$\frac{3}{2}$ColoringAlg1 vs. GreedyD$\frac{3}{2}$ColoringAlg2: (different number of colors)

4.4 Experimental Results

Our experimental work in the Hessian estimation case has two objectives: (i) to experimentally demonstrate the advantage of exploiting symmetry, and (ii) to compare and contrast the performance of the two distance-$\frac{3}{2}$ coloring algorithms.

4.4.1 Test Graphs

The matrices behind the test graphs used in our experiments arise from finite element methods [14]. Table 4 gives some structural information about the adjacency graphs of these matrices.

Graph	$\|V\|$	$\|E\|$	Δ	δ	$\overline{\delta}$
mrng1	257000	505048	4	2	3
mrng2	1017253	2015714	4	2	3
598a	110971	741934	26	5	13
144	144649	1074393	26	4	14
m14b	214765	1679018	40	4	15
auto	448695	3314611	37	4	14

Table 4: Graph statistics

Graph	$\chi(d2)$	$\chi(d\frac{3}{2})_{alg1}$	$\chi(d\frac{3}{2})_{alg2}$	$T(d2)$	$T(d\frac{3}{2})_{alg1}$	$T(d\frac{3}{2})_{alg2}$
mrng1	12	8	10	0.37	0.61	0.35
mrng2	12	9	10	1.74	2.90	1.75
598a	38	27	32	0.74	3.00	0.75
144	41	28	35	0.90	4.35	0.93
m14b	42	29	34	1.19	5.65	1.23
auto	42	29	36	3.97	17.4	4.07
Total	187	130	157	8.91	33.91	9.08

Table 5: Results of $d2$-coloring and $d\frac{3}{2}$-coloring

4.4.2 Results and Discussion

Table 5 shows the performance of Algorithms GreedyD2Coloring, GreedyD$\frac{3}{2}$-ColoringAlg1 and GreedyD$\frac{3}{2}$ColoringAlg2 when applied on our test graphs. The left-half of the table shows the number of colors used by the different algorithms and the right-half shows the corresponding time (in seconds) spent on coloring.

The results clearly demonstrate the advantage of exploiting symmetry: distance-$\frac{3}{2}$ coloring requires significantly fewer colors than distance-2 coloring. The total number of colors required by GreedyD$\frac{3}{2}$ColoringAlg1 overall involved test graphs is about 30% less than the amount required by GreedyD2Coloring. The table also shows nicely the time/quality trade-off between the two distance-$\frac{3}{2}$ coloring algorithms. GreedyD$\frac{3}{2}$ColoringAlg2 uses nearly the same time as GreedyD2Coloring while the number of colors required is 16% less.

5 Direct Estimation of Jacobians Using Bidirectional Partitions

5.1 Why Use Bidirectional Partition?

We call a partition that involves both the rows and columns of a matrix a *bidirectional* partition as opposed to a *unidirectional* partition in which either only rows or only columns are involved. We motivate the need for a bidirectional partition within the context of automatic differentiation (AD).

AD is a chain rule based technique for evaluating the derivatives of functions defined by computer programs. It has two basic modes of operation known as *forward* and *reverse*. These modes correspond to a bottom-up and a top-down strategy of accumulating partial derivatives of elementary functions that define the computational scheme of the function to be differentiated. A treatment of the technical details of AD is beyond the scope of this paper, but the interested reader is referred to, for instance, the books [10, 16, 17].

What is of interest for us is that, as in the FD setting, the efficient computation of matrices using AD gives rise to partitioning problems in which structural orthogonality continues to be the partition-criterion. In particular, one can use the forward mode to compute a group of columns of a matrix A from the product Ad_1, where the vector d_1 has nonzeros in positions corresponding to columns of A that are structurally orthogonal. Furthermore, in the reverse mode, a group of structurally orthogonal rows of A can be computed from the vector-matrix product ${d_2}^T A$, where d_2 is an appropriately defined vector. This means that one can potentially take advantage of the sparsity available in columns and in rows.

One way of exploiting sparsity in columns and rows is to *separately* partition the columns and rows of the matrix and use the partition which gives the minimum number of groups. For a symmetric matrix, a row partition is equivalent to a column partition, but for a nonsymmetric matrix, the two partitions may differ considerably. For example, consider an $n \times n$ matrix where all the entries on the diagonal and the first row are nonzero, and the rest of the matrix entries are all zero. For such a matrix structure, a column partition requires n groups whereas a row partition requires just two groups.

However, an approach based on a separate row and column partition is not always satisfactory. For example, consider an $n \times n$ matrix where all of the elements in the first row, first column, and the diagonal are nonzero and the rest of the entries are all zero. For such a structure, a row partition requires n groups and so does a column partition. However, using a *combined* row and column partition, three groups are enough to determine all the nonzero entries of the matrix. First, separately evaluate the entries in the first column and the first row (two groups). Then, since the remaining $(n-1) \times (n-1)$ matrix is diagonal, determine all entries by one evaluation. Thus, three groups (two column and one row) suffice to determine all the nonzero entries.

In Section 5.2 we consider such a computation of a nonsymmetric matrix using the combined modes of AD via a direct method. (The corresponding problem using a substitution method will be discussed in Section 6.2)

Note that a bidirectional partition does not make sense for computing a symmetric matrix. In particular, a symmetry-exploiting unidirectional partition is sufficient.

5.2 The Matrix Partitioning Problem

Consider the vectors $d_1, d_2, \ldots, d_p$ in Problem 3.1 as the p columns of the $n \times p$ matrix D. In the automatic differentiation literature, the matrix D is known as a *seed matrix*. An alternative way of posing Problem 3.1 would then be: given the structure of a matrix $A \in R^{m\times n}$, find a seed matrix $D \in R^{n\times p}$ with the least value of p such that the product AD determines A directly. By the same token, the problem that arises in the bidirectional efficient direct computation of a Jacobian can be posed as follows.

Problem 5.1 *Given the sparsity structure of the matrix $A \in R^{m\times n}$, find matrices $D_1 \in R^{n\times p_1}$ and $D_2 \in R^{m\times p_2}$ such that AD_1 and $D_2^T A$ together determine A directly and the value $p = p_1 + p_2$ is minimized.*

Hossain and Steihaug [20] studied Problem 5.1 and reformulated it as a partitioning problem by using the notion of *consistent row-column partition* in which the *entire* set of rows and columns is partitioned into two respective set of groups. Coleman and Verma [9] also studied the same problem and identified a similar bidirectional partition problem. Their notion of partition differs from that of Hossain and Steihaug in that it partitions only a *subset* of the columns and the rows of the matrix that suffice for the direct determination of the entries. The following concepts were used to formalize the requirements.

Definition 5.2 A *bipartition* of a matrix A is a pair (Π_C, Π_R) where Π_C is a column partition of a subset of the columns of A and Π_R is a row partition of a subset of the rows of A.

Definition 5.3 A bipartition (Π_C, Π_R) of a matrix A is *consistent with a direct determination* if for every nonzero entry a_{ij} of A, either column j is in a group of Π_C which has no other column having a nonzero in row i, or row i is in a group of Π_R which has no other row having a nonzero in column j.

The number of column and row groups in a consistent bipartition corresponds to the number of forward and reverse AD passes, respectively, required to compute the nonzero entries directly. To see this, observe that a nonzero a_{ij} can be determined either from a column group where column j is the only column with a nonzero in row i, or from a row group where row i is the only row with a nonzero in column j.

Figure 16 shows an example of a bipartition of a matrix consistent with a direct determination. In the example, Π_C includes columns 1 and 3 whereas Π_R includes rows 2, 4, and 5. As can be seen from the color used at the left

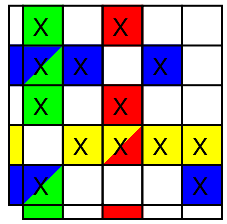


Figure 16: A consistent bipartition of a matrix. The color(s) used on each nonzero entry shows the column and/or row group from which it can be computed.

and bottom edges of the figure, columns 1 forms one column group and column 3 forms another; similarly, rows 2 and 5 form one group and row 4 forms a second row group. Thus, the bipartition uses a total of four groups. Notice that some entries of the matrix (painted with one color) can be computed only from one group while others (painted with two colors) can be computed from either of two groups. As the reader can easily verify, for this example, a row-only or a column-only consistent partition would require five groups.

Assuming that the computational costs involved in the forward and reverse modes of AD are of the same order, in an efficient method based on using a bipartition (Π_C, Π_R), the value $|\Pi_C| + |\Pi_R|$ is required to be as small as possible. Thus Problem 5.1 can be stated as a partitioning problem in the following way.

Problem 5.4 (*MPP3*) *Given the sparsity structure of a matrix $A \in R^{m\times n}$, find a bipartition (Π_C, Π_R) of A consistent with a direct determination such that $|\Pi_C| + |\Pi_R|$ is minimized.*

It should be pointed out that in the formulation of MPP3 we are concerned only with *computational* cost. However, in general, the forward mode of AD requires less memory space than the reverse mode making the former more desirable. Hence, a more accurate objective in this context would be to minimize $w_1|\Pi_C| + w_2|\Pi_R|$ for some empirically determined *weights* w_1 and w_2[3].

5.3 The Graph Formulation

When a nonsymmetric matrix A is represented by its bipartite graph $G_b(A) = (V_1, V_2, E)$, we have shown that a column partition consistent with a direct determination can be obtained by finding a partial distance-2 coloring of

[3]Paul Hovland pointed out the possibility of considering a weighted version to us.

G_b on V_2. Let us consider how this coloring needs to be modified to find a bipartition consistent with a direct determination. Notice that the coloring we are looking for has to meet the following conditions.

- The sets V_1 and V_2 should use disjoint set of colors.
- Some vertices may not be involved in the determination of any nonzero entry of the underlying matrix. Such vertices are assigned a 'neutral' color (say color zero).
- Since every nonzero matrix entry has to be determined, for every edge in E, at least one of the endpoints has to be assigned a nonzero color.
- A nonzero matrix entry may be determined either from a positively colored column vertex or a positively colored row vertex. This suggests that the coloring condition sought here is some relaxation of the distance-2 coloring requirement imposed in the case of unidirectional partition.

The following definition, introduced by Coleman and Verma [9], albeit using a different terminology, formalizes the conditions listed above. The subsequent theorem establishes the equivalence between the matrix and graph problems.

Definition 5.5 Let $G_b = (V_1, V_2, E)$ be a bipartite graph. A mapping $\phi : [V_1, V_2] \rightarrow \{0, 1, \ldots, p\}$ is a *distance-$\frac{3}{2}$ bicoloring* of G_b if the following conditions hold.

1. If $u \in V_1$ and $v \in V_2$, then $\phi(u) \neq \phi(v)$ or $\phi(u) = \phi(v) = 0$.
2. If $(u, v) \in E$, then $\phi(u) \neq 0$ or $\phi(v) \neq 0$.
3. If vertices u and v are adjacent to vertex w with $\phi(w) = 0$, then $\phi(u) \neq \phi(v)$.
4. Every path of three edges uses at least three colors.

Figure 17 shows the matrix of Figure 16 together with its bicoloring representation.

Theorem 5.6 [Coleman and Verma[9]]
Let A be an $m \times n$ matrix and $G_b(A) = (V_1, V_2, E)$ be its bipartite graph. The mapping $\phi : [V_1, V_2] \rightarrow \{0, 1, \ldots, p\}$ is a distance-$\frac{3}{2}$ p-bicoloring if and only if ϕ induces a bipartition (Π_C, Π_R) of A, with $|\Pi_C| + |\Pi_R| = p$, consistent with a direct determination.

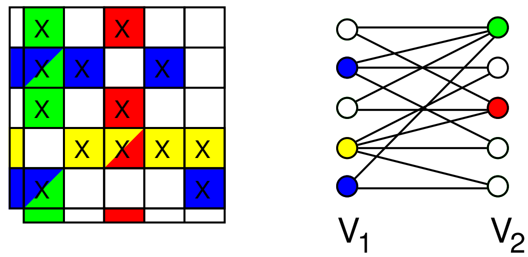


Figure 17: A consistent bipartition and its representation as a distance-$\frac{3}{2}$ p-bicoloring

Thus MPP3 is equivalent to the following graph problem.

Problem 5.7 (*GCP3*) *Given the bipartite graph $G_b(A) = (V_1, V_2, E)$ representing the sparsity structure of an $m \times n$ matrix A, find a distance-$\frac{3}{2}$ p-bicoloring of $G_b(A)$ with the least value of p.*

5.4 A Distance-$\frac{3}{2}$ Bicoloring Algorithm

In a distance-$\frac{3}{2}$ p-bicoloring some vertices are assigned the neutral color 0. For example, see the uncolored vertices in Figure 17. We make the following crucial observation which helps us identify a possible set of such vertices. The observation is a direct consequence of Condition 2 of Definition 5.5.

Observation 5.8 *Let $G_b = (V_1, V_2, E)$ be a bipartite graph and $\phi : [V_1, V_2] \to \{0, 1, \ldots, p\}$ be a distance-$\frac{3}{2}$ bicoloring of G_b. Then,*

- *the set $C = \{v : \phi(v) \neq 0\}$ is a **vertex cover** in G_b, and*
- *the set $I = \{v : \phi(v) = 0\}$ is an **independent set** in G_b.*

One consequence of Observation 5.8 is that $|I| + |C| = |V_1| + |V_2|$. Thus, a decrease in the cardinality of the vertex cover C results in an increase in the cardinality of the independent set I.

In terms of solving coloring problem GCP3, Observation 5.8 suggests the scheme outlined in GCP3Algorithm.

In Step 1 of GCP3Algorithm any vertex cover can be used. However, the choice of the vertex cover affects the subsequent coloring in Step 3, both in terms of number of colors used and coloring time spent. To reduce the coloring time in Step 3, the size of the vertex cover should be minimized. On the other hand, minimizing the potential number of colors to be used

GCP3Algorithm($G_b = (V_1, V_2, E)$)

1. Find a vertex cover C in G_b.
2. Assign the vertices in the set $I = (V_1 \cup V_2) \setminus C$ the color 0.
3. Color the vertices in C such that the result is a distance-3/2 bicoloring of G_b.

imposes an additional requirement: the vertex cover should include those vertices from V_1 and V_2 with relatively high number of distance-1 neighbors. From the discussion in Section 5.1, it is to be recalled that matrices with a few dense rows and columns benefit from a bidirectional partition.

In a bipartite graph, a minimum cardinality vertex cover can be obtained via finding a *maximum matching* in polynomial time [27, 35]. In fact, it can be computed practically in effectively linear time in the number of edges [29].

Once Steps 1 and 2 are carried out, Step 3 can be done by a suitable adaptation of GreedyD$\frac{3}{2}$ColoringAlg1. GreedyD$\frac{3}{2}$BiColoring, given here only pictorially, is such an adaptation. One of the differences between the coloring and bicoloring algorithms is that in the latter case, two disjoint set of colors are used in coloring the vertices in V_1 and V_2 of the bipartite graph $G_b = (V_1, V_2, E)$. Another difference is that at a step of the bicoloring algorithm where v is colored, a vertex within the distance-3 neighborhood of v may be one of *three* types: it is colored with a positive value, it is colored with 0, or it is not yet colored. The choice of color for v thus needs to consider these three options.

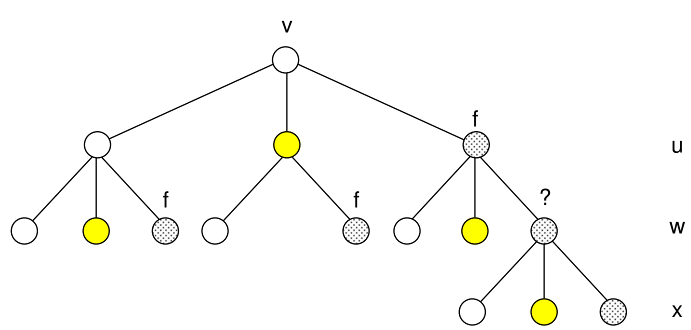


Figure 18: Visualizing a step in GreedyD$\frac{3}{2}$BiColoring

Figure 18 shows a visual presentation of the ith step of GreedyD$\frac{3}{2}$BiColoring. Note the similarity with Figure 13. Notice also that, in choosing a color for vertex v in $G_b = (V_1, V_2, E)$, since the colors for vertices in V_1 and V_2 are drawn from two disjoint sets, we need only consider colors of vertices two edges away from v. In the figure, shaded nodes correspond to colored vertices, yellow nodes show vertices with color zero, and unshaded nodes correspond to uncolored vertices. Observe that the node with color 0 at level u has only two children. The colors of the vertices in the nodes marked by an f indicate forbidden colors and whether the color at the node marked by '?' is forbidden or not depends on the color used at node x: if $\phi(u) = \phi(x)$, $\phi(w)$ is forbidden, otherwise, it is not.

The time complexity of GreedyD$\frac{3}{2}$BiColoring is $O((|V_1| + |V_2|)\overline{\delta}_3)$, which is also the overall time complexity of GCP3Algorithm assuming that step 1 is done using a greedy algorithm that is linear in the number of edges.

Notice that a partial distance-2 coloring of G_b on V_2 is a just special case of the scheme GCP3Algorithm. To see this, consider the trivial choice of vertex cover $C = V_2$ in Step 1. This implies that, in Step 2, the vertices in the set $I = V_1$ will be colored with zero. By Condition 3 of Definition 5.5, vertices adjacent to a vertex colored with zero are required to be assigned different colors. Thus, the result is effectively a partial distance-2 coloring of G_b on V_2.

We note that Hossain and Steihaug [20] and Coleman and Verma [9] have each proposed an algorithm for GCP3. These algorithms can be interpreted in light of GCP3Algorithm. The algorithm of Hossain and Steihaug (HS) *implicitly* finds a vertex cover while the coloring of the graph proceeds. Using our terminology, the vertices that remain uncolored at the end of the HS-algorithm form an independent set in the graph and can thus be assigned the neutral color 0.

The algorithm of Coleman and Verma uses a preprocessing step to identify the rows and columns of the underlying matrix that eventually need to be colored with positive values. The preprocessing step uses a matrix-based procedure which effectively produces a small sized vertex cover. After the preprocessing step, a certain 'column intersection' graph, adapted to the distance-$\frac{3}{2}$ bicoloring requirements, is constructed to finally use known distance-1 coloring heuristics on the resulting graph.

5.5 Other Graph Problems in Automatic Differentiation

In the context of AD, exploiting sparsity, when available, is one way by which the computation of Jacobians can be made efficient. There do exist other ways of optimizing Jacobian computation. For example, the accumulation of Jacobian matrices by *elimination* methods on the associated computational graph is an interesting recent approach [30]. Depending on the type of the elimination technique employed, these methods result in a number of combinatorial problems that could be regarded as searching for shortest paths in graphs. Examples of studies that investigate such methods can be found in [10, 16, 17, 30, 31].

6 Substitution Methods

In a unidirectional partition based estimation of a matrix A via a substitution method, the binary vectors $d_1, \ldots, d_p$ are chosen such that the system of equations defined by the products $Ad_1, \ldots, Ad_p$ becomes triangular and hence the unknowns can be determined via substitution. Such a partition needs to fulfill a more relaxed set of requirements compared to one used for a direct method and hence results in smaller number of groups. This fact has been especially used when estimating a symmetric matrix since substitution can be effectively combined with the exploitation of symmetry [4]. For a nonsymmetric matrix, the advantage offered by a substitution method over a direct method while using unidirectional partitions is not so pronounced. (An example of a variant of a substitution method for a nonsymmetric matrix estimation using a unidirectional partition can be found in [21].) On the other hand, one may get some advantage in using bidirectional partitions suitable for substitution methods in nonsymmetric matrix estimation in comparison with bidirectional partitions suitable for direct methods.

In the rest of this section, within the context of substitution methods, we consider unidirectional partitions for symmetric matrices (Section 6.1) and bidirectional partitions for nonsymmetric matrices (Section 6.2).

6.1 Estimating Hessians

To illustrate the fact that a partition used in a substitution method requires fewer groups than that used in a direct method, consider the 4×4 symmetric matrix H shown in Figure 19.

For matrix H, any partition consistent with a direct determination requires at least three groups. One such partition is $\{(h_1, h_4), (h_2), (h_3)\}$, where h_j is the jth column of H. However, if we do not insist on determining the elements directly, two groups would suffice. For example $\{(h_1, h_3), (h_2, h_4)\}$. The two matrix-vector products corresponding to the two groups yield a system of eight equations involving the nonzero entries of H. Note that, due to symmetry, nonzero element h_{ij} can be identified with h_{ji}, and hence, there are effectively seven unknowns in the system. This system can be ordered to

$$\begin{bmatrix} X & X & & \\ X & X & X & \\ & X & X & X \\ & & X & X \end{bmatrix}$$

Figure 19: The structure of a 4×4 matrix

a triangular form and be solved via substitution.

6.1.1 The Matrix Partitioning Problem

In general, a partition of the columns of a symmetric matrix induces a substitution method if there is an ordering of the matrix unknowns such that all unknowns can be solved for, in that order, using symmetry and previously solved elements.

We formally define such a partition and then state the corresponding partitioning problem.

Definition 6.1 A partition of the columns of a symmetric matrix A is said to be *substitutable* if there exists an ordering on the elements of A such that for every nonzero a_{ij}, either a_j is in a group where all the nonzeros in row i, from other columns in the same group, are ordered before a_{ij} or a_i is in a group where all the nonzeros in row j, from other columns in the same group, are ordered before a_{ij}.

Problem 6.2 (*MPP4*) *Given the sparsity structure of a symmetric matrix $A \in R^{n \times n}$, find a substitutable partition of the columns of A that has the least number of groups.*

6.1.2 The Graph Formulation

To formulate MPP4 as a graph problem, we introduce the notion of *acyclic* coloring. Coleman and Cai [4] established the connection between acyclic[4] coloring and the estimation of a symmetric matrix using a substitution method. Acyclic coloring had been studied earlier by Grünbaum [18] in a different context. Some more references to works on acyclic coloring can be found in the book [23].

[4]Coleman and Cai use the term cyclic coloring to refer to what is known as acyclic coloring in the graph theoretic literature.

Definition 6.3 *A mapping $\phi : V \to \{1, 2, \ldots, p\}$ is an acyclic p-coloring of a graph $G = (V, E)$ if ϕ is distance-1 p-coloring and every cycle in G uses at least three colors.*

Theorem 6.4 **[Coleman and Cai [4]]**
Let A be a symmetric matrix with nonzero diagonal elements and $G(A) = (V, E)$ be its adjacency graph representation. A mapping ϕ is an acyclic p-coloring of $G(A)$ if and only if ϕ induces a substitutable partition of the columns of A.

Consider the 5×5 symmetric matrix and its adjacency graph depicted in Figure 20. An optimal acyclic coloring that uses three colors is shown. It can be verified that this induces a substitutable partition of the columns. By contrast, a symmetric partition consistent with a direct determination (a distance-$\frac{3}{2}$ coloring) would have required four groups (colors).

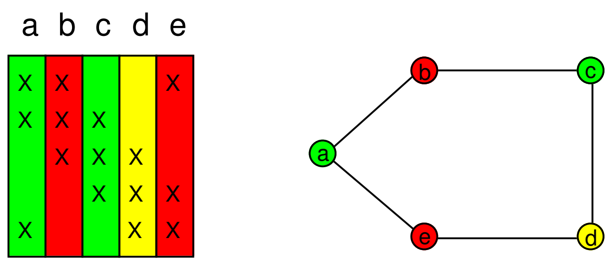


Figure 20: A 5×5 symmetric matrix and an acyclic coloring of its adj. graph

In view of Theorem 6.4, MPP4 is equivalent to the following graph problem.

Problem 6.5 **(*GCP4*)** *Given the adjacency graph $G(A) = (V, E)$ representing the sparsity structure of a symmetric matrix $A \in R^{n \times n}$ with nonzero diagonal elements, find an acyclic p-coloring of $G(A)$ with the least value of p.*

6.1.3 An Acyclic Coloring Algorithm

The problem of finding an acyclic coloring of a graph using the the least possible number of colors is NP-hard [4]. Here we present an effective algorithm called GreedyAcyclicColoring to find an approximate solution.

The basic idea in our algorithm is to detect and 'break' a two-colored cycle while an otherwise distance-1 coloring of the graph proceeds. Specifically, the algorithm colors the vertices of a graph $G = (V, E)$ while making a Depth

First Search (DFS) traversal. Recall that a *back-edge* in the *DFS-tree* of an undirected graph defines a unique cycle. For more information on DFS, refer to the book [11].

In GreedyAcyclicColoring, the *DFS-tree* $T(G)$ of the graph G is implicitly constructed as the algorithm proceeds. Let $\phi(v)$ denote the color of vertex v and $s(v)$ denote the order in which v is first visited in the DFS traversal of G $(1 \leq s(v) \leq |V|)$. The root r of $T(G)$ has $s(r) = 1$. Further, let $p(v)$ be a pointer to the *parent* of v in $T(G)$, and $l(v)$ be a pointer to the lowest ancestor of v in $T(G)$ such that $\phi(l(v)) \neq \phi(v)$ and $\phi(l(v)) \neq \phi(p(v))$. The latter pointer will be used in the detection of a two-colored cycle. In particular, the algorithm proceeds in such a way that the path in $T(G)$ from v up to, but not including, $l(v)$ is two-colored. At the beginning of the algorithm, for every vertex u, $l(u)$ is set to point to null.

Consider the step in GreedyAcyclicColoring where vertex v is first visited. To start with, v is assigned the smallest color different from all of its distance-1 neighbors in G, including its parent $p(v)$ in $T(G)$. If there exists a back-edge $b = (v, w)$ in the current $T(G)$ such that $s(l(p(v))) < s(w)$ and $\phi(v) = \phi(p(p(v)))$, then this implies that the cycle corresponding to b is two-colored (see Figure 21 which shows a partial view of the DFS-tree at the step where vertex v is to be colored). To break the cycle, v is assigned a new color—the smallest color different from $\phi(p(v))$ and $\phi(p(p(v)))$—and $l(v)$ is set to point to $p(p(v))$. Otherwise, if no such back-edge exists, the color of v is declared final and $l(v)$ is updated in the following manner. If $\phi(v) \neq \phi(p(p(v)))$, then $l(v) = p(p(v))$; otherwise $l(v) = l(p(v))$.

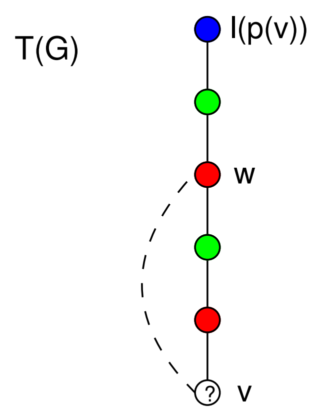


Figure 21: Visualizing a step in GreedyAcyclicColoring

Notice that the amount of work done in each DFS visit of a vertex v in the graph is proportional to the degree-1 of v. This makes GreedyAcyclicColoring an $O(|E|)$-time algorithm.

We note that Coleman and Cai [4] have proposed an algorithm for the acyclic coloring problem. The idea in their algorithm is to first transform a given graph $G = (V, E)$ to a 'completed' graph $G' = (V, E')$ such that a distance-1 coloring of G' is equivalent to an acyclic coloring of G, and then use a known distance-1 coloring heuristic on G'. The construction of G' is done in the following way. Start by setting $E' = E$; visit the vertices in V in a *predefined order*; at each step i, if vertex v_i is adjacent to vertices v_j and v_k such that both v_j and v_k are ordered before v_i then add the edge (v_j, v_k) to E'.

Our approach differs from that of Coleman and Cai in at least two ways. First, the graph G' used in the latter approach may require substantially more storage space than the original graph G used in our approach. Second, an edge in $E' \setminus E$ in the latter approach may actually be redundant. For example, a distance-1 coloring of an odd-length cycle in G uses at least three colors and hence is a valid acyclic coloring whereas the Coleman and Cai approach adds one redundant edge to the cycle.

6.2 Estimating Jacobians

In estimating a nonsymmetric matrix using a substitution method, the requirement on the bipartition can be relaxed so as to obtain fewer number of groups in comparison with a direct method.

6.2.1 The Matrix Partitioning Problem

We state the following definition used by Coleman and Verma [9] to subsequently give the fifth matrix partitioning problem of our concern.

Definition 6.6 A bipartition (Π_C, Π_R) of a matrix A is *consistent with a determination by substitution* if there exists an ordering on the elements of A such that for every nonzero entry a_{ij}, either column j is in a group where all nonzeros in row i, from other columns in the group, are ordered before a_{ij} or row i is in a group where all the nonzeros in column j, from other rows in the group, are ordered before a_{ij}.

Problem 6.7 (*MPP5*) *Given the sparsity structure of a matrix $A \in R^{m \times n}$, find a bipartition (Π_C, Π_R) of A consistent with a determination by substitution such that $|\Pi_C| + |\Pi_R|$ is minimized.*

6.2.2 The Graph Formulation

The relationship between bicoloring and bipartition, established by Theorem 5.6, coupled with that between acyclic coloring and substitutable partition, established by Theorem 6.4, suggests that 'acyclic bicoloring' might be the right graph model for MPP5. Coleman and Verma [9] showed that this was indeed the case.

Definition 6.8 Let $G_b = (V_1, V_2, E)$ be a bipartite graph. A mapping $\phi : [V_1, V_2] \to \{0, 1, \ldots, p\}$ is an *acyclic bicoloring* of G_b if the following conditions hold.

1. If $u \in V_1$ and $v \in V_2$, then $\phi(u) \neq \phi(v)$ or $\phi(u) = \phi(v) = 0$.
2. If $(u, v) \in E$, then $\phi(u) \neq 0$ or $\phi(v) \neq 0$.
3. If vertices u and v are adjacent to vertex w with $\phi(w) = 0$, then $\phi(u) \neq \phi(v)$.
4. Every cycle uses at least three colors.

Theorem 6.9 [Coleman and Verma [9]]
Let A be an $m \times n$ matrix and $G_b(A) = (V_1, V_2, E)$ be its bipartite graph. The mapping $\phi : [V_1, V_2] \to \{0, 1, \ldots, p\}$ is an acyclic p-bicoloring if and only if ϕ induces a bipartition (Π_C, Π_R) of A, with $|\Pi_C| + |\Pi_R| = p$, consistent with determination by substitution.

By Theorem 6.9, the following coloring problem is equivalent to MPP5.

Problem 6.10 (*GCP5*) *Given the bipartite graph $G_b(A) = (V_1, V_2, E)$ representing the sparsity structure of an $m \times n$ matrix A, find an acyclic p-bicoloring of $G_b(A)$ with the least value of p.*

6.2.3 An Acyclic Bicoloring Algorithm

By virtue of Observation 5.8, the approach we suggest for solving the acyclic bicoloring problem is given in GCP5Algorithm.

GCP5Algorithm($G_b = (V_1, V_2, E)$)

1. Find a vertex cover C in G_b.
2. Assign the vertices in the set $I = (V_1 \cup V_2) \setminus C$ the color 0.
3. Color the vertices in C such that the result is an acyclic bicoloring of G_b.

Matrix	UniPartition	BiPartition	Method
Jacobian	distance-2 coloring	distance-$\frac{3}{2}$ bicoloring	Direct
Hessian	distance-$\frac{3}{2}$ coloring	NA	Direct
Jacobian	NA	acyclic bicoloring	Substitution
Hessian	acyclic coloring	NA	Substitution

Table 6: Graph coloring formulations for estimating *all* nonzero entries of derivative matrices. The Jacobian and the Hessian are represented by their bipartite and adjacency graphs, respectively. NA stands for not applicable.

7 Summary of Full Matrix Estimation Problems

7.1 Summary

Table 6 summarizes the graph coloring problems that arise in efficient derivative matrix estimation, and particularly, in the case where *all* nonzero entries of a matrix are required to be computed. Note that the problems in Table 6 are formulated independent of the numerical technique used. For example, the distance-2 coloring problem could arise in using finite differences or only the forward mode of automatic differentiation. Furthermore, althogh bipartition problems have previously been studied in the context of AD, it is also possible to use FD techniques in conjunction with bipartitions.

In this section we expose the inter-relationship among the various coloring variants introduced so far. In doing so, we show that distance-2 coloring is the most *general* variant.

The conditions required by distance-1 coloring, acyclic coloring, distance-$\frac{3}{2}$ coloring, acyclic bicoloring, distance-$\frac{3}{2}$ bicoloring, and distance-2 coloring imply the following relationships among their respective chromatic numbers. The chromatic numbers for distance-k coloring (and bicoloring) of a general graph G (and bipartite graph G_b) are denoted by $\chi_k(G)$ (and $\chi_{kb}(G_b)$). Similarly the chromatic numbers for acyclic coloring (and bicoloring) of a graph G (and a bipartite graph G_b) are denoted by $\chi^a(G)$ (and $\chi^{ab}(G_b)$).

Theorem 7.1 *For a general graph $G = (V, E)$,*

$$\chi_1(G) \leq \chi^a(G) \leq \chi_{\frac{3}{2}}(G) \leq \chi_2(G) = \chi_1(G^2).$$

Proof: Observe that a distance-2 coloring is a distance-$\frac{3}{2}$ coloring; a distance-$\frac{3}{2}$ coloring is an acyclic coloring; and an acyclic coloring is a distance-1 coloring. □

Theorem 7.2 *For a bipartite graph $G_b = (V_1, V_2, E)$,*

$$\chi^{ab}(G_b) \leq \chi_{\frac{3}{2}b}(G_b) \leq min\{\chi_1({G_b}^2[V_1]), \chi_1({G_b}^2[V_2])\},$$

where $({G_b}^2[W])$ is the sub-graph of ${G_b}^2$ induced by W.

Proof: The first inequality is obvious. For the second inequality, observe that a partial distance-2 coloring on V_2 is a valid distance-$\frac{3}{2}$ bicoloring where all the vertices in V_1 are specified to be colored with 0. A similar argument, with the roles of V_1 and V_2 interchanged, can be used to complete the proof. $\square$

In the context of matrix estimation using numerical methods, the implication of Theorem 7.2 is that, an optimal bidirectional partition, irrespective of the structure of the matrix, yields fewer (or at most as many) groups compared to an optimal unidirectional partition, and hence potentially results in a more efficient computation.

Theorems 7.1 and 7.2 show that distance-2 coloring is an archetypal model in the estimation of Jacobian and Hessian matrices using techniques that rely on unidirectional partition via direct and substitution methods.

Distance-2 coloring has other applications. Examples include channel assignment [25] and facility location problems (see Chapter 5 in [34]). From a more theoretical perspective, distance-2 coloring for planar graphs has been studied in [2] and a similar study for chordal graphs is available in [1].

7.2 Other Methods

The matrix estimation methods considered in this paper relied on unidirectional or bidirectional *partitions* that treat a row or a column of a matrix as an *'atomic'* entity. Moreover, the partitions were required to be *consistent* (or symmetrically consistent) with a direct or a substitution based determination of the matrix entries. However, approaches where one or more of these underlying principles are not necessarily followed have also been suggested. Such approaches may lead to fewer groups but often entail very high prices.

For example, Newsam and Ramsdell [32] proposed an *indirect* method that enables the determination of the entries of an $m \times n$ Jacobian matrix A using ρ_{max} groups where ρ_{max} is the maximum number of nonzeros in a row of A. The method is optimal in terms of the number of groups used. However, it needs to solve n least squares problems; besides the resulting system can be ill-conditioned, making the method far less practical. Note the distinction between indirect methods as used in this context and substitution methods as used in the rest of the paper. Indirect methods refer in general to cases where a matrix entry is not directly read off from a matrix-vector product but is rather extracted by some other means. Substitution methods, on the other hand, rely on solving an underlying (essentially) triangular system of equations.

In the same work, Newsam and Ramsdell also suggest a method that generalizes consistent column partitioning. They use the terms *variable isolation* (VI) to refer to methods based on consistent column partitioning and *element isolation* (EI) to refer to the generalized method. In a matrix A, element (entry) a_{ij} is said to be isolated from a_{pq} whenever $a_{iq} = a_{pj} = 0$ or $j = q$. The EI-method partitions the nonzero matrix entries into as few groups consisting of pairwise isolated elements as possible. The element partition is then used to determine a column grouping (that allows a group to contain structurally non-orthogonal columns and/or columns to reside in several groups) that enables the determination of all nonzero matrix entries. Newsam and Ramsdell also formulate the EI partitioning problem as a coloring problem on a graph where the vertices are the nonzero entries of the matrix and an edge between two entries exists whenever they are not isolated from each other. Clearly, a VI-method implies an EI-method but not vice-versa. Thus an EI-method would use at most as many groups as a VI-method. The EI-method is obviously impractical as it deals with 'extremely refined' partitioning (the graph that needs to be colored involves nnz vertices and $O((nnz)^2)$ edges, where nnz is the number of nonzero entries in the matrix!).

Hossain and Steihaug [19, 22] suggest a generalized framework that has EI and VI as its special cases. In particular, they suggest a technique for direct estimation of an $m \times n$ Jacobian in which the rows are first grouped into l blocks that define 'segmented' columns and then the segments are partitioned into groups each of which consists of structurally independent segments. Note that $l = 1$ corresponds to a VI-method and $l = m$ corresponds to an EI-method. Hossain and Steihaug show that, for some matrix structures, there exists an $l \neq 1$ that results in fewer groups compared with an approach based on $l = 1$. In a recent report [22] they characterize the nature of a partition that leads to an optimal number of groups. The optimal partition is also formulated as a coloring of an associated graph.

In the case of symmetric matrix estimation, McCormick [28] gives a classification of direct methods, including those that do not necessarily rely on consistent partitions.

8 Partial Matrix Estimation

In many PDE constrained optimization contexts, the Jacobian or the Hessian is formed only for preconditioning purposes. For preconditioning, it is often common to compute a *subset* of the matrix elements. Computing a good preconditioner is critical for fast convergence to the solution. The recent survey article [24] discusses various applications where methods known as "Jacobian-free Newton-Krylov" are used. A basic ingredient in the use of these methods is an approximate computation of *some* elements of the Jacobian. Also, there are examples in which only certain elements of the Hessian need to be updated in an iterative procedure, while others do not because they are unlikely to change in value [3].

In this section we develop the graph coloring formulations of the partitioning problems that arise in the estimation of a *specified subset* of the nonzero entries of a matrix via direct methods. We call this *partial* matrix estimation as opposed to *full* matrix estimation, where all nonzero entries are required to be determined.

The coloring formulations in this section are new and more sophisticated than the coloring formulations in full matrix estimation. The motivation for developing the new graph formulations is that efficient partial matrix estimation can be used to further reduce the number of colors needed to estimate the required elements. For example, if only the diagonal elements of a Hessian are needed, then we need only the *distance-1 coloring* of the adjacency graph, rather than the distance-$\frac{3}{2}$ coloring required for full matrix estimation.

The colorings defined in this section allow a vertex to have the color zero. A vertex with color zero signifies the fact that it would not be used to estimate any element of the matrix represented by the graph, i.e., columns or rows that correspond to the color zero are not used to estimate any elements in those columns or rows.

The rest of this section is organized in three parts. Each part deals with a scenario defined by the kind of matrix under consideration (symmetric or nonsymmetric) and the type of partition employed (unidirectional or bidirectional). In each case, the required entries are assumed to be determined using a direct method. The problems that correspond to estimation via substitution are not considered in this paper.

8.1 Nonsymmetric Matrix, Unidirectional Partition

Let $A \in R^{m \times n}$ be a nonsymmetric matrix, and S denote the set of nonzero elements of A required to be estimated. A partition $\{C_1, \ldots, C_p\}$ of a subset of the columns of A is *consistent with a direct determination of* S if for every $a_{ij} \in S$, column a_j is included in some group that contains no other column with a nonzero in row i.

Let $G_b(A) = (V_1, V_2, E)$ be the bipartite graph of A, and $F \subseteq E$ correspond to the elements of S. A mapping $\phi : V_2 \to \{0, 1, \ldots, p\}$ is a *distance-2 coloring of* G_b *restricted to* F if the following conditions hold for every $(v, w) \in F$, where $v \in V_1$, $w \in V_2$.

1. $\phi(w) \neq 0$, and
2. for every path (u, v, w), $\phi(u) \neq \phi(w)$.

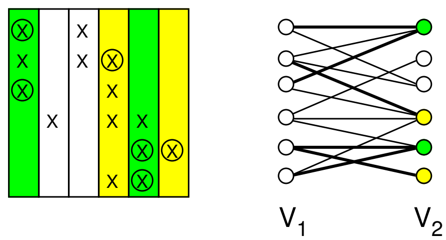


Figure 22: Partial estimation of a nonsymmetric matrix

Theorem 8.1 *The mapping* ϕ *is a distance-*2 p*-coloring of* $G_b(A)$ *restricted to* F *if and only if* ϕ *induces a column partition* $\{C_1, \ldots, C_p\}$ *consistent with a direct determination of* S*.*

Proof: Assume that ϕ is a distance-2 p-coloring of $G_b(A)$ restricted to F. We show that the groups $\{C_1, \ldots, C_p\}$ where $C_\alpha = \{a_j : \phi(a_j) = \alpha\}$, $1 \leq \alpha \leq p$, constitute a partition of a subset of the columns of A consistent with a direct determination of S. First, observe that by coloring condition 1, for every $a_{ij} \in S$ ($(r_i, a_j) \in F$), $\phi(a_j) \neq 0$ and thus column a_j belongs to group $C_{\phi(a_j)}$ and hence is involved in the partition. Assume now that the partition induced by the coloring is not consistent with a direct determination of S. This occurs only if there exist nonzero elements a_{ij} and a_{ik}, $k \neq j$, such that $a_{ij} \in S$ and both a_j and a_k belong to group $C_{\alpha'}$ for some α', $1 \leq \alpha' \leq p$. But this contradicts coloring condition 2, and hence cannot occur.

Conversely, assume that the partition $C = \{C_1, \ldots, C_p\}$ is consistent with a direct determination of S. Construct a coloring ϕ of $G_b(A)$ as follows. $\phi(a_j) = \alpha$ if $a_j \in C_\alpha$, and $\phi(a_j) = 0$ if $a_j \notin C$. We claim that ϕ is a distance-2 p-coloring of $G_b(A)$ restricted to F. Each vertex in V_2 incident to an edge in F corresponds to a column with an entry in S and thus gets a nonzero color.

Thus ϕ satisfies coloring condition 1. Consider any path (a_j, r_i, a_k) where $(r_i, a_j) \in F$. Note that such a path in $G_b(A)$ exists whenever entries a_{ij} and a_{ik} are nonzero. The partition condition implies that column a_k cannot be in the same group as a_j. Thus, by construction, $\phi(a_j) \neq \phi(a_k)$, satisfying coloring condition 2. □

8.2 Symmetric Matrix, Unidirectional Partition

Let $A \in R^{n \times n}$ be a symmetric matrix with nonzero diagonal elements, and S denote the set of nonzero elements of A required to be estimated. A partition $\{C_1, \ldots, C_p\}$ of a subset of the columns of A is *symmetrically consistent with a direct determination of* S if for every $a_{ij} \in S$ at least one of the following two conditions are met.

1. The group containing a_j has no other column with a nonzero in row i.
2. The group containing a_i has no other column with a nonzero in row j.

Let $G(A) = (V, E)$ be the adjacency graph of A, $F_{od} \subseteq E$ correspond to the off-diagonal elements in S, and F_d correspond to the diagonal elements in S, i.e., $F_d = \{(u, u) : u \in U\}$ where $U \subseteq V$. Let $F = F_{od} \cup F_d$. A mapping $\phi : V \to \{0, 1, 2, \ldots, p\}$ is a *distance-$\frac{3}{2}$ coloring of G restricted to F* if the following conditions hold.

1. For every $(u, u) \in F_d$,
   - 1.1. $\phi(u) \neq 0$, and
   - 1.2. for every $(u, v) \in E$, $\phi(u) \neq \phi(v)$.
2. For every $(v, w) \in F_{od}$,
   - 2.1. $\phi(v) \neq \phi(w)$, and
   - 2.2. at least one of the following two conditions holds:
   - 2.2.1. $\phi(v) \neq 0$ and for every path (v, w, x), $\phi(v) \neq \phi(x)$ or
   - 2.2.2. $\phi(w) \neq 0$ and for every path (u, v, w), $\phi(u) \neq \phi(w)$.

Theorem 8.2 *The mapping ϕ is a distance-$\frac{3}{2}$ p-coloring of $G(A)$ restricted to F if and only if ϕ induces a column partition $\{C_1, \ldots, C_p\}$ symmetrically consistent with a direct determination of S.*

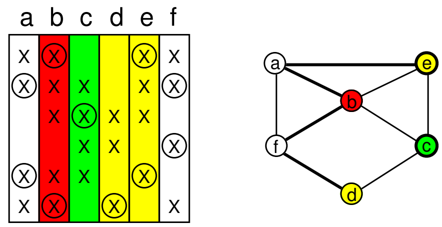


Figure 23: Partial estimation of a symmetric matrix

Proof: Assume that ϕ is a distance-$\frac{3}{2}$ p-coloring of $G(A)$ restricted to F. We show that the groups $\{C_1, \ldots, C_p\}$ where $C_\alpha = \{a_j : \phi(a_j) = \alpha\}$, $1 \leq \alpha \leq p$, constitute a partition of a subset of the columns of A symmetrically consistent with a direct determination of S.

By coloring conditions 1 and 2, for every $a_{ij} \in S$ ($(a_i, a_j) \in F$), at least one of a_i or a_j has a nonzero color and hence is involved in the partition $\{C_1, \ldots, C_p\}$. Let $a_{ij} \in S$ be a diagonal entry ($i = j$). Then, coloring condition 1 ensures that $\phi(a_i) \neq 0$ and that $\phi(a_i) \neq \phi(a_k)$ for every $(a_i, a_k) \in E$. Thus, by construction, column a_i belongs to group $C_{\phi(a_i)}$ and no column a_k, $k \neq i$ with $a_{ik} \neq 0$ is in $C_{\phi(a_i)}$. This clearly satisfies the partition condition. Let $a_{ij} \in S$ now be an off-diagonal entry ($i \neq j$). Assume without loss of generality that $\phi(a_j) \neq 0$. By condition 2.1, $\phi(a_j) \neq \phi(a_i)$. By condition 2.2.2, there is no path (a_k, a_i, a_j) in $G(A)$, for any $k \neq i, j$ such that $\phi(a_j) = \phi(a_k)$. The last two statements together imply that column a_j belongs to group $C_{\phi(a_j)}$ and that no column a_k, $k \neq j$ with $a_{ik} \neq 0$ is in $C_{\phi(a_j)}$. This satisfies partition condition 1. A similar argument applies to the case where $\phi(a_i) \neq 0$ which implies the satisfaction of the alternative partition condition.

To prove the converse, assume that the partition $C = \{C_1, \ldots, C_p\}$ is symmetrically consistent with a direct determination of S. Construct a coloring ϕ of $G(A)$ as follows. $\phi(a_j) = \alpha$ if $a_j \in C_\alpha$, and $\phi(a_j) = 0$ if $a_j \notin C$. We claim that ϕ is a distance-$\frac{3}{2}$ p-coloring of $G(A)$ restricted to F.

Consider a diagonal element $a_{ii} \in S$. The partition conditions ensure that a_i is in some group $C_{\alpha'}$ and that there is no column $a_k \in C_{\alpha'}$, $k \neq i$ such that $a_{ik} \neq 0$. Thus, by construction, $\phi(a_i) \neq 0$ and $\phi(a_i) \neq \phi(a_k)$ for every $(a_i, a_k) \in E$, satisfying coloring condition 1. Consider now the case where $a_{ij} \in S$ is an off-diagonal element. First, observe that since all diagonal elements are nonzero, a_i and a_j cannot belong to the same group. Thus $\phi(a_i) \neq \phi(a_j)$, satisfying coloring condition 2.1. Second, observe that there are two possibilities by which the partitioning conditions have been satisfied. We consider only one of these; the second one can be treated in a similar manner. Suppose a_j belongs to some group $C_{\alpha'}$ and that there is no other column $a_k \in C_{\alpha'}$, $k \neq j$ such that $a_{ik} \neq 0$. Thus, by construction, $\phi(a_j) \neq 0$ and $\phi(a_j) \neq \phi(a_k)$ for every path (a_k, a_i, a_j) in $G(A)$, satisfying coloring condition 2.2.2. □

A special case of Theorem 23 is the problem of estimating *only* the diagonal elements of A, i.e., $F_d = \{(v, v) : v \in V\}$ and $F_{od} = \emptyset$. For this problem, condition 1 is the only applicable condition, and states that a distance-1 coloring of $G(A)$ is sufficient.

8.3 Nonsymmetric Matrix, Bidirectional Partition

Let $A \in R^{m \times n}$ be a nonsymmetric matrix, and S denote the set of nonzero elements of A required to be estimated. A bipartition (Π_C, Π_R) of a subset of the columns and rows of A is *consistent with a direct determination of* S if for every $a_{ij} \in S$ at least one of the following conditions are met.

1. The group (in Π_C) containing column j has no other column having a nonzero in row i.
2. The group (in Π_R) containing row i has no other row having a nonzero in column j.

Let $G_b(A) = (V_1, V_2, E)$, and $F \subseteq E$ correspond to the elements in S. A mapping $\phi : [V_1, V_2] \rightarrow \{0, 1, \ldots, p\}$ is said to be a *distance-$\frac{3}{2}$ bicoloring of G_b restricted to F* if the following conditions are met.

1. Vertices in V_1 and V_2 receive disjoint colors, except for color 0; i.e., for every $u \in V_1$ and $v \in V_2$, either $\phi(u) \neq \phi(v)$ or $\phi(u) = \phi(v) = 0$.
2. At least one endpoint of an edge in F receives a nonzero color; i.e., for every $(v, w) \in F$, $\phi(v) \neq 0$ or $\phi(w) \neq 0$.
3. For every $(v, w) \in F$,
   - 3.1. if $\phi(v) = 0$, then, for every path (u, v, w), $\phi(u) \neq \phi(w)$,
   - 3.2. if $\phi(w) = 0$, then, for every path (v, w, x), $\phi(v) \neq \phi(x)$,
   - 3.3. if $\phi(v) \neq 0$ and $\phi(w) \neq 0$, then for every path (u, v, w, x), either $\phi(u) \neq \phi(w)$ or $\phi(v) \neq \phi(x)$.

Theorem 8.3 *The mapping ϕ is a distance-$\frac{3}{2}$ p-bicoloring of G_b restricted to F if and only if ϕ induces a bipartition (Π_C, Π_R), $|\Pi_C| + |\Pi_R| = p$, consistent with a direct determination of S.*

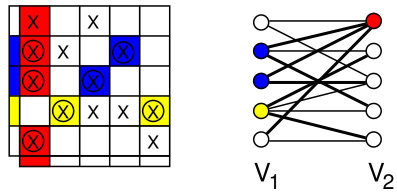


Figure 24: Partial estimation of a nonsymmetric matrix using bipartition

Proof: Let the construction of a partition given a coloring, and vice-versa, be done in a similar manner as in the proof of Theorem 8.1.

Assume that ϕ is a distance-$\frac{3}{2}$ bicoloring of $G_b(A)$ restricted to F. Let the induced bipartition be (Π_C, Π_R). Coloring condition 1 implies that (Π_C, Π_R) is a bipartition. By condition 2, for every $a_{ij} \in S$, either $a_j \in \Pi_C$ or $r_i \in \Pi_R$ (or both). Assume now that (Π_C, Π_R) is not consistent with a direct determination of S. This occurs only if one of the following cases hold for any $a_{ij} \in S$:

- $\phi(r_i) = 0, \phi(a_j) \neq 0$ and there exists a column a_k, $k \neq j$ with $a_{ik} \neq 0$ such that $\phi(a_j) = \phi(a_k)$. But this contradicts coloring condition 3.1, and hence cannot occur.

- $\phi(a_j) = 0, \phi(r_i) \neq 0$ and there exists a row r_l, $l \neq i$ with $a_{lj} \neq 0$ such that $\phi(r_i) = \phi(r_l)$. But this contradicts coloring condition 3.2, and hence cannot occur.

- $\phi(r_i) \neq 0, \phi(a_j) \neq 0$, and there exist column a_k, $k \neq j$ with $a_{ik} \neq 0$, and row r_l, $l \neq i$ with $a_{lj} \neq 0$ such that $\phi(a_j) = \phi(a_k)$ and $\phi(r_i) = \phi(r_l)$. But this contradicts coloring condition 3.3, and hence cannot occur.

Hence, (Π_C, Π_R) is consistent with a direct determination of S.

Conversely, assume that (Π_C, Π_R) is a bipartition consistent with a direct determination of S. Clearly, the constructed coloring ϕ satisfies conditions 1 and 2. To complete the proof, we show that ϕ also satisfies condition 3. Assume that ϕ violates condition 3. Then one of the following cases must have happened:

- There exists a path (a_k, r_i, a_j) for some $(r_i, a_j) \in F$ such that $\phi(r_i) = 0$ and $\phi(a_j) = \phi(a_k)$. But this implies that element a_{ij} cannot be determined directly, contradicting the assumption that (Π_C, Π_R) is consistent with a direct determination of S.

- There exists a path (r_i, a_j, r_l) for some $(r_i, a_j) \in F$ such that $\phi(a_j) = 0$ and $\phi(r_i) = \phi(r_l)$. Again this implies that element a_{ij} cannot be determined directly, a contradiction of our assumption.

- There exists a path (a_k, r_i, a_j, r_l) for some $(r_i, a_j) \in F$ such that $\phi(r_i) = \phi(r_l) \neq 0$ and $\phi(a_j) = \phi(a_k) \neq 0$. But this implies that element a_{ij} cannot be determined directly, contradicting the assumption.

$\square$

9 Conclusion

We have studied the efficient estimation of sparse Jacobian and Hessian matrices using FD and AD techniques. We considered methods that rely on a unidirectional as well as a bidirectional partition to be used in an evaluation based on a direct or a substitution scheme. We introduced partial matrix estimation problems in distinction from full matrix estimation problems. In doing so, we developed a unified graph theoretic framework to cope with a variety of complex matrix partitioning problems.

At the basis of our graph problem formulations lies a robust graph representation of the sparsity structure of a matrix: a nonsymmetric matrix is represented by its bipartite graph and a symmetric matrix by its adjacency graph.

We showed that the distance-2 graph coloring problem is a generic model for the various unidirectional matrix partitioning problems.

Our unified graph theoretic approach enabled us to provide some fresh insight into the matrix problems and as a result we developed several simple and effective algorithms. Among others, we developed a new linear-time heuristic algorithm for the acyclic coloring problem. We showed experimentally the advantages offered by our partial distance-2 coloring formulation as compared to a distance-1 coloring formulation for the case of nonsymmetric matrix determination via a direct method using unidirectional partition. In the case of symmetric matrix evaluation using a direct method, we demonstrated a time/quality trade-off between two distance-$\frac{3}{2}$ coloring algorithms.

Our emphasis in this work has been on greedy algorithms. Other algorithmic techniques need to be explored in the future. Furthermore, it could be interesting to find a distance-2 coloring algorithm that uses asymptotically the same time as the greedy algorithm discussed in this paper and balances the number of vertices in each color class. Finding a random color, rather than the smallest color, from an allowable set could be an idea to consider in this regard.

In the case of two-dimensional partition problems, based on the known relationship to graph bicoloring, we argued that finding a 'small'-size vertex cover as a preprocessing step contributes to making the overall computation more efficient. A more precise characterization of the 'optimum' vertex cover required is a worthwhile issue.

We have not developed any special algorithms for the restricted coloring problems that arise in partial matrix estimation. We think, however, that the ideas used in our algorithms for the coloring problems in full matrix estimation can be adapted to the restricted cases by observing the particular coloring conditions.

In general, most of the algorithms in the literature for solving the coloring problems considered in this paper rely on first transforming the input graph $G = (V, E)$ to some denser graph $G' = (V, E')$, $E' \supseteq E$, such that a distance-1 coloring of G' is equivalent to the particular coloring problem on G. In contrast, the algorithms proposed in this paper solve the particular coloring problem directly on G. As has been argued, the main advantages offered by our approach are the possibility to mix-and-match methods, less storage space requirement, and ease of developing flexible software.

One of the motivations for the current study has been the need for the development of parallel algorithms for solving partitioning problems in large-scale PDE-constrained optimization contexts. In a recent work [15], we have shown some parallel algorithms (using the shared-memory programming model) for the distance-2 and distance-$\frac{3}{2}$ coloring problems. Our results, theoretical as well as experimental, were promising. We believe that the current study lays a foundation for further work on the development and implementation of not only shared-memory but also distributed-memory parallel algorithms.

Acknowledgements

References

[1] G. Agnarsson, R. Greenlaw, and M. M. Halldórsson. On powers of chordal graphs and their colorings. *Congress Numerantium*, 100:41–65, 2000.

[2] G. Agnarsson and M. M. Halldórsson. Coloring powers of planar graphs. *SIAM J. Discrete Math.*, 16(4):651–662, 2003.

[3] O. Axelsson and U. Nävert. On a graphical package for nonlinear partial differential equation problems. In B. Gilchrist, editor, *Proceedings of IFIP Congress 77*, Information Processing, pages 103–108. North-Holland, 1977.

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

[5] T. F. Coleman, B. Garbow, and J. J. Moré. Software for estimating sparse Jacobian matrices. *ACM Trans. Mathematical Software*, 10:329–347, 1984.

[6] T. F. Coleman, B. Garbow, and J. J. Moré. Software for estimating sparse Hessian matrices. *ACM Trans. Mathematical Software*, 11:363–377, 1985.

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

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

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

[10] G. Corliss, C. Faure, A. Griewank, L. Hascoét, and U. Naumann. *Automatic Differentiation of Algorithms: From Simulation to Optimization*. Springer, New York, 2002.

[11] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. Massachusetts Institute of Technology, 1990.

[12] 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.

[13] T. Davis. University of Florida Sparse Matrix Collection. *NA Digest*, 97(23), June 7 1997. http://www.cise.ufl.edu/research/sparse/matrices ftp://ftp.cise.ufl.edu/pub/faculty/davis/matrices.

[14] A.H. Gebremedhin and F. Manne. Scalable parallel graph coloring algorithms. *Concurrency: Practice and Experience*, 12:1131–1146, 2000.

[15] A.H. Gebremedhin, F. Manne, and A. Pothen. Parallel distance-k coloring algorithms for numerical optimization. In B. Monien and R. Feldmann, editors, *Euro-Par 2002 Parallel Processing*, volume 2400 of *LNCS*, pages 912–921. Springer Verlag, 2002.

[16] A. Griewank. *Evaluating Derivatives. Principles and Techniques of Algorithmic Differentiation.* SIAM, Philadelphia, 2000.

[17] A. Griewank and G. F. Corliss. *Automatic Differentiation of Algorithms: Theory, Implementation, and Application.* SIAM, Philadelphia, 1991.

[18] B. Grünbaum. Acyclic colorings of planar graphs. *Israel J. Math.*, 14:390–408, 1973.

[19] S. Hossain. On the computation of sparse Jacobian matrices and Newton steps. Technical Report 146, Department of Informatics, University of Bergen, Norway, March 1998.

[20] S. Hossain and T. Steihaug. Computing a sparse Jacobian matrix by rows and columns. *Optimization Methods and Software*, 10:33–48, 1998.

[21] S. Hossain and T. Steihaug. Reducing the number of AD passes for computing a sparse Jacobian matrix. In G. Corliss, C. Faure, A. Griewank, L. Hascoét, and U. Naumann, editors, *Authomatic Differentiation of Algorithms: From Simulation to Optimization*, pages 263 – 270. Springer, 2002.

[22] S. Hossain and T. Steihaug. Optimal direct determination of sparse jacobian matrices. Technical Report 254, Department of Informatics, University of Bergen, Norway, October 2003.

[23] T. Jensen and B. Toft. *Graph Coloring Problems.* Wiley-Interscience Publications, New York, 1995.

[24] D. A. Knoll and D. E. Keyes. Jacobian-free Newton-Krylov methods: A survey of appoaches and applications. Available at http://www.math.odu.edu/~ keyes/.

[25] S. O. Krumke, M. V. Marathe, and S.S. Ravi. Approximation algorithms for channel assignment in radio networks. In *Dial M for Mobility*, Dallas, Texas, October 1998.

[26] Y. Lin and S. S. Skiena. Algorithms for square roots of graphs. *SIAM J. Disc. Math.*, 8:99–118, 1995.

[27] L. Lovász and M.D. Plummer. *Matching Theory.* North-Holland, Amsterdam, 1986.

[28] S. T. McCormick. Optimal approximation of sparse Hessians and its equivalence to a graph coloring problem. *Math. Program.*, 26:153–171, 1983.

[29] R. Motwani. Average-case analysis of algorithms for matchings and related problems. *Journal of the ACM*, 41:1329–1356, 1994.

[30] U. Naumann. Optimal accumulation of Jacobian matrices by elimination methods on the dual computational graph. *Math. Prog.* Manuscript.

[31] U. Naumann. Cheaper Jacobians by simulated annealing. *SIAM J. Opt.*, 3(13):660–674, March 2002.

[32] G. N. Newsam and J. D. Ramsdell. Estimation of sparse Jacobian matrices. *SIAM J. Alg. Disc. Meth.*, 4:404–418, 1983.

[33] M. J. D. Powell and P. L. Toint. On the estimation of sparse Hessian matrices. *SIAM J. Numer. Anal.*, 16(6):1060–1074, December 1979.

[34] V. V. Vazirani. *Approximation Algorithms.* Springer, 2001.

[35] L. A. Wolsey. *Integer Programming.* Wiley-Interscience Publication, New York, USA, 1998.