A NEW PIVOTING STRATEGY FOR GAUSSIAN ELIMINATION

Markus Olschowka
Institut für Angewandte Mathematik
Universität Freiburg
D-79104 Freiburg
Germany

and

Arnold Neumaier
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974-0636
U.S.A.
email: neum@research.att.com

December 1993

Abstract.
This paper discusses a method for determining a good pivoting sequence for Gaussian elimination, based on an algorithm for solving assignment problems. The worst case complexity is $O(n^3)$, while in practice $O(n^{2.25})$ operations are sufficient.

1991 MSC Classification: 65F05 (secondary 90B80)

Keywords: Gaussian elimination, scaled partial pivoting, I-matrix, dominant transversal, assignment problem, bipartite weighted matching.

1 Introduction

For a system of linear equations $Ax = b$ with a square nonsingular coefficient matrix A, the most important solution algorithm is the systematic elimination method of Gauss. The basic idea of Gaussian elimination is the factorization of A as the product LU of a lower triangular matrix L with ones on its diagonal and an upper triangular matrix U, the diagonal entries of which are called the pivot elements. In general, the numerical stability of triangularization is guaranteed only if the matrix A is symmetric positive definite, diagonally dominant, or an H-matrix.

In [4], p.159, CONTE & DEBOOR write: It is "difficult (if not impossible) to ascertain for a general linear system how various pivoting strategies affect the accuracy of the computed solution. A notable and important exception to this statement are the linear systems with positive definite coefficient matrix (...). For such a system, the error in the computed solution due to rounding errors during elimination and back-substitution can be shown to be acceptably small if the trivial pivoting strategy of no interchanges is used. But it is not possible at present to give a "best" pivoting strategy for a general linear system, nor is it even clear what such a term might mean. (...) A currently accepted strategy is *scaled partial pivoting*".

Partial pivoting consists in choosing – when the kth variable is to be eliminated – as pivot element the element of largest absolute value in the remainder of the kth column and exchanging the corresponding rows. For good numerical stability it is advisable to carry out the partial pivoting with prior scaling of the coefficient matrix A – i.e., replacing of A by D_1AD_2 , where D_1 and D_2 are nonsingular diagonal matrices. To avoid additional rounding errors the scaling is done implicitly: Let $A = (a_{ij})$ be an $n{\times}n$- matrix and $D_1 = \text{Diag}(d_1^1, ..., d_n^1)$, then, in the jth reduction step $A^{(j-1)} \to A^{(j)}$, the pivoting rule becomes:
Find $k \geq j$ with

$$|a_{kj}^{(j-1)}| \cdot d_k^1 = \max_{i \geq j} |a_{ij}^{(j-1)}| \cdot d_i^1 \neq 0,$$

and then take $a_{kj}^{(j-1)}$ as pivot element.
The nonsingular diagonal matrix D_1 must be chosen such that the triangularization with partial pivoting of D_1AD_2 is stable. (The matrix D_2 does not affect the pivot choice, but is introduced for convenience.) The approach

to this problem taken by WILKINSON [13] is to insist that the scaled matrix be *equilibrated*, i.e., all its rows have l_1-norm 1. This can be achieved for arbitrary choice of D_2, since the diagonal matrix D_1 whose diagonal entries are the inverse norms of the rows of AD_2 is the unique diagonal matrix which makes D_1AD_2 equilibrated. The choice of D_2 then determines D_1 which in turn determines the pivot elements. In practice one usually takes for D_2 the identity matrix, $D_2 = I$. However, this can be disastrous when the entries in some row of A have widely differing magnitudes, and it is particularly unsatisfactory for sparse matrices (CURTIS & REID [5]).
One can improve the behavior by enforcing also that the transposed matrix is equilibrated, i.e., that all columns have norm 1. Ignoring the signs, the scaled matrix is then *doubly stochastic*, i.e., the entries of each row and column sum to 1. Now the scaling matrices must satisfy nonlinear equations which are not easy to solve. PARLETT & LANDIS describe in [11] iterative procedures for scaling nonnegative matrices to double stochastic form, and they present results of tests comparing the new algorithms with other methods. However, their examples suggest that their iterations require $O(n^4)$ operations, so these procedures are too slow to be useful as a scaling strategy for Gaussian elimination.

In the present paper we introduce a new idea for choosing the scaling matrices in such a way that $\tilde{A} = D_1AD_2$ has a structure resembling an equilibrated, diagonally dominant matrix. Thus, choosing D_1 as the scaling matrix for implicit partial pivoting, we expect better results in Gaussian elimination than with the traditional choices.
More precisely, we set ourselves the following task: Given the matrix A, find diagonal matrices D_1 and D_2 such that $\tilde{A} = D_1AD_2$ has the following form:

(*) All coefficients are of absolute value at most 1 and there are n elements of absolute value 1, no two of which lie in the same row or column.

In the special case when A is obtained by scaling and permuting a diagonally dominant matrix A_0 it turns out that the triangular factorization of A using partial pivoting and implicit scaling with scaling factors determined by (*) chooses a pivot sequence equivalent to original ordering of A_0. During reduction those elements are taken as pivots, whose corresponding entries in $\tilde{A}$ are of absolute value 1. *Thus the condition (*) recovers in this case the natural stable ordering.* The same happens for scaled and permuted symmetric positive definite matrices. This nourishes the hope that we also get a good pivoting sequence in the general situation. Although we cannot prove this, the experimental results obtained in Section 4 indeed suggest that this holds.

In Section 2 we investigate the properties of matrices with property (*) and of corresponding scaling matrices. In Section 3 we discuss in more detail the consequences for pivoting. Section 4 describe the particular method we implemented to produce scaling factors such that the scaled matrix satisfies (*). The method involves at most $O(n^3)$ operations, i.e., the worst case work is of the same order as for the subsequent triangular factorization. Test results with some random matrices are given in Section 5.

Acknowledgment. I'd like to thank Carl de Boor and an unknown referee for a number of useful remarks which helped to improve this paper.

2 Theoretical Foundations

Let A be an $n{\times}n$ - matrix and denote by $\Sigma := Sym(n)$ the set of permutations of $1, ..., n$. For $\sigma \in \Sigma$, P_σ denotes the $n{\times}n$ - permutation matrix with entries

$$p_{ij} := \begin{cases} 1 & \text{if } j = \sigma(i), \\ 0 & \text{otherwise.} \end{cases}$$

$P_\sigma A$ is obtained from A by permuting its rows in such a way that the elements $a_{\sigma(j),j}$ are on its diagonal.

2.1. Lemma. *If A is nonsingular, there exists at least one transversal for A.*

Proof: Since

$$\det(A) := \sum_{\sigma\in\Sigma} \operatorname{sgn}(\sigma) \prod_{j=1}^{n} a_{\sigma(j),j} \neq 0$$

there exists some $\sigma \in \Sigma$ with $\prod_{j=1}^{n} a_{\sigma(j),j} \neq 0$. □

We call a matrix A *structurally nonsingular* if A has a transversal. A useful condition for this can be deduced from the

2.2. Marriage Theorem. (Philip Hall, cf. Bose & Manvel [1]):
Let N be the set of columns indices of A and $M(J) := \{i \mid \exists j \in J : a_{ij} \neq 0\}$ for $J \subseteq N$. Then there exists a transversal for A iff

$$|M(J)| \geq |J| \text{ for all } J \subseteq N. \qquad \square$$

Now, writing the condition as

$$n - |M(J)| + |J| \leq n \text{ for all } J \subseteq N,$$

we get:

2.3. Proposition. *A is structurally nonsingular iff $m_1 + m_2 < n$ for all $m_1 \times m_2$ - submatrices containing zeros.* $\square$

We call the permutation $\pi \in \Sigma$ a *transversal* for A if

$$a_{\pi(j),j} \neq 0 \text{ for all } j = 1, ..., n.$$

The transversal is called *dominant* if

$$\prod_{j=1}^{n} |a_{\pi(j),j}| = \max_{\sigma \in \Sigma} \prod_{j=1}^{n} |a_{\sigma(j),j}|.$$

The next lemma shows the invariance of the dominant transversal under scaling.

2.4. Lemma. *Let D_1 and D_2 be nonsingular diagonal matrices. Then π is a dominant transversal for A iff π is a dominant transversal for $\tilde{A} = D_1 A D_2$.*

Proof: Let π be a dominant transversal for A. Since $D_1 := \operatorname{diag}(d_1^1, ..., d_n^1)$ and $D_2 := \operatorname{diag}(d_1^2, ..., d_n^2)$ are nonsingular,

$$\tilde{a}_{\pi(j),j} = d^1_{\pi(j)} a_{\pi(j),j} d^2_j \neq 0 \text{ for all } j = 1, ..., n.$$

Given $\sigma \in \Sigma$, then, since τ and π are bijective,

$$\prod_{j=1}^{n} \frac{|\tilde{a}_{\sigma(j),j}|}{|\tilde{a}_{\pi(j),j}|} = \prod_{j=1}^{n} \frac{|d^1_{\sigma(j)} a_{\sigma(j),j} d^2_j|}{|d^1_{\pi(j)} a_{\pi(j),j} d^2_j|} = \prod_{j=1}^{n} \frac{|a_{\sigma(j),j}|}{|a_{\pi(j),j}|} \leq 1.$$

Since $A = D_1^{-1} \tilde{A} D_2^{-1}$, the converse direction holds, too. $\square$

A $n{\times}n$ - matrix A is called an *H-Matrix* if

$$\|I - D_1 A D_2\|_\infty < 1 \tag{1}$$

for suitable diagonal matrices D_1 and D_2. (The special case where one can take for D_2 the identity defines *strictly diagonally dominant* matrices.)

2.5. Theorem. *For H-matrices and symmetric positive definite matrices, $\pi = id$ is the unique dominant transversal.*

Proof: (i) In (1), D_1 and D_2 are nonsingular matrices, because

$$|1 - d^1_i a_{ii} d^2_i| \leq |1 - d^1_i a_{ii} d^2_i| + \sum_{\substack{k=1 \\ k \neq i}}^{n} |d^1_i a_{ik} d^2_k| < 1$$

implies $d^1_i \neq 0, \;\; d^2_i \neq 0$. According to Lemma 2.4 it only remains to show that $\pi = id$ is the unique dominant transversal for any matrix $\tilde{A} = D_1 A D_2$ satisfying $\|I - \tilde{A}\|_\infty < 1$. But such an $\tilde{A}$ is strongly diagonally dominant, hence this is true.

(ii) Let A be symmetric positive definite. Then

$$a_{ii} > 0 \;\; \text{for } i = 1, ..., n$$

since $e^{(i)T} A e^{(i)} > 0$ where $e^{(i)}$ is the ith unit vector in $\mathbb{R}^n$. Defining $D_1 = D_2 = \mathrm{diag}(a_{11}^{-1/2}, ..., a_{nn}^{-1/2})$, the matrix $\tilde{A} = D_1 A D_2$ is still symmetric positive definite with units on its diagonal. Again with Lemma 2.4 we must only show that id is dominant for $\tilde{A}$.

Let $x^{(l,m)} := e^{(l)} + e^{(m)}$ and

$$y^{(l,m)} := e^{(l)} - e^{(m)} \quad \text{for } l, m = 1, ..., n, \; l \neq m.$$

Then

$$\begin{aligned} x^{(l,m)T}\tilde{A}x^{(l,m)} &= \tilde{a}_{ll}+\tilde{a}_{lm}+\tilde{a}_{ml}+\tilde{a}_{mm} \\ &= 2(1+\tilde{a}_{lm}) > 0 \end{aligned}$$

and similarly

$$y^{(l,m)T}\tilde{A}x^{(l,m)} = 2(1-\tilde{a}_{lm}) > 0.$$

Thus

$$|\tilde{a}_{lm}| < 1 \quad \text{for } l,m = 1,...,n,\ l \neq m,$$

so that any transversal $\pi \neq id$ gives a product $< 1 = \prod a_{ii}$. □

2.6. Corollary. *Permuting a matrix such that a dominant transversal is on the diagonal restores a symmetric positive definite matrix, a strongly diagonally dominant matrix, or an H-matrix from any permuted version of it.*

We call a $n\times n$ - matrix A an *I-matrix* if

$$|a_{ii}| = 1 \text{ for } i = 1,...,n \tag{2}$$

and

$$|a_{ij}| \leq 1 \text{ for } i,j = 1,...,n,\ i \neq j, \tag{3}$$

and a *strong I-matrix* if (2) and

$$|a_{ij}| < 1 \text{ for } i,j = 1,...,n,\ i \neq j. \tag{4}$$

It is clear that $\pi = id$ is dominant for I-matrices and diagonally dominant matrices. Moreover, when A is a strong I-matrix or strongly diagonally dominant then no other transversal can be dominant, too.

2.7. Example. Rice [12] described the difficulties to scale the matrix

$$A = \begin{pmatrix} 1 & 10^{+20} & 10^{+10} & 1 \\ 10^{+20} & 10^{+20} & 1 & 10^{+40} \\ 10^{+10} & 1 & 10^{+40} & 10^{+50} \\ 1 & 10^{+40} & 10^{+50} & 1 \end{pmatrix}$$

to an equilibrated matrix. In our setting there is a dominant transversal corresponding to the permutation matrix

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Using the scaling matrices

$$\begin{aligned} D_1 &= \operatorname{diag}(1, 10^{-15}, 10^{-20}, 10^{-25}), \\ D_2 &= \operatorname{diag}(10^{-5}, 10^{-20}, 10^{-25}, 10^{-30}), \end{aligned}$$

we get the I-matrix

$$PD_1AD_2 = \begin{pmatrix} 1 & 10^{-15} & 10^{-40} & 10^{-5} \\ 10^{-5} & 1 & 10^{-15} & 10^{-30} \\ 10^{-30} & 10^{-5} & 1 & 10^{-55} \\ 10^{-15} & 10^{-40} & 10^{-5} & 1 \end{pmatrix}$$

which is a strongly diagonal dominant matrix and thus perfectly scaled for Gaussian elimination.

Now we are prepared to solve our task described in the introduction by the following theorem:

2.8. Theorem. *For every structurally nonsingular $n \times n$ - matrix A there are a permutation matrix P and two diagonal matrices D_1, D_2 such that PD_1AD_2 is an I-matrix. Moreover, P, D_1, D_2 can be found in $O(n^3)$ operations.*
(In practice usually $O(n^{2.25})$ operations suffice for full matrices).

We prove the theorem constructively by finding a dominant transversal, permuting it to the diagonal, and scaling it appropriately. We need some notation. Let

$$\mathbb{R}_\infty := \mathbb{R} \cup \{-\infty, +\infty\}.$$

If

$$\bar{a}_i := \max_{j=1,\dots,n} |a_{ij}| \neq 0 \quad \text{for all } i = 1, ..., n,$$

then the matrix $C = (c_{ij}) \in \mathbb{R}_\infty^{n\times n}$ is defined as follows:

$$c_{ij} := \begin{cases} \log \bar{a}_i - \log |a_{ij}| & \text{if } a_{ij} \neq 0, \\ +\infty & \text{otherwise.} \end{cases}$$

2.9. Lemma. *A transversal π for A is dominant iff $\sum\limits_{j=1}^{n} c_{\pi(j),j}$ is minimal in C among all $\pi \in \Sigma$ with $c_{\pi(j),j} < \infty$ for $j = 1, ..., n$.*

Proof: Since

$$\sum_{j=1}^{n} c_{\pi(j),j} = \log \prod_{j=1}^{n} \frac{\bar{a}_{\pi(j)}}{|a_{\pi(j),j}|} ,$$

it follows that

$$\sum_{j=1}^{n} c_{\pi(j),j} \text{ is minimal for all } \sigma \in \Sigma \text{ with } c_{\sigma(j),j} < \infty$$

$$\Leftrightarrow \prod_{j=1}^{n} \frac{\bar{a}_{\pi(j)}}{|a_{\pi(j),j}|} \text{ is minimal over all transversals for } A$$

$$\Leftrightarrow \prod_{j=1}^{n} \frac{|a_{\pi(j),j}|}{\bar{a}_{\pi(j)}} \text{ is maximal over all transversals for } A$$

$$\Leftrightarrow \pi \text{ is a dominant transversal for } A. \quad \square$$

Lemma 2.9 shows that a dominant transversal can be found for the matrix C by searching for a permutation π which minimizes

$$\sum_{j=1}^{n} c_{\pi(j),j}.$$

This problem is known as the *assignment problem* (AP) or *bipartite weighted matching problem.* The classical $O(n^3)$-algorithm for AP is the so-called *Hungarian method* (KUHN [10]), but later some faster methods have been proposed, cf. Section 4.

Lemma 2.4 tells us in which cases there is no transversal for A. If there are only zeros in one row i then the matrix C cannot be built because the maximal row element $\bar{a}_i$ does not exist. In the other cases the matching algorithm cannot compute an admissible permutation because for all $\sigma \in \Sigma$ there is a column j with $c_{\sigma(j),j} = \infty$. (In all these cases A is singular (Lemma 2.1) and therefore unsuitable for Gaussian elimination; so this does not restrict applicability.)

Usually the assignment problem is formulated in graph theoretical terminology. Let $G = (V_1, V_2, E)$ be a bipartite graph with V_1 and V_2 denoting the bipartition of the nodes. An edge $e \in E$ will be written as an ordered pair (i, j) with $i \in V_1$ and $j \in V_2$. G is called *complete (bipartite)*, if $(i, j) \in E$ for all $i \in V_1$ and $j \in V_2$. A *matching (perfect matching)* in G is a subset M of the edges such that every node in $V_1 \cup V_2$ is incident to at most one (exactly one) edge in M. Given weights $c_{ij} \in \mathbb{R}$ for the edges $(i, j) \in E$ we can associate a weight

$$c(M) = \sum_{(i,j) \in M} c_{ij}$$

with every matching M. With $G = (V_1, V_2, E)$, $V_1 = V_2 = \{1, ..., n\}$ and $E = \{(i, j) \mid c_{ij} < \infty\}$ the assignment problem is equivalent to the problem of finding a perfect matching with minimal weight.

It is well-known that the assignment problem can be written as the following linear program:

(LP) Find variables $x_{ij} \in \mathbb{R},\ i \in V_1,\ j \in V_2$ minimizing

$$\sum_{(i,j) \in E} c_{ij} x_{ij}$$

under the conditions

$$\sum_{\substack{j \in V_2 \\ (i,j) \in E}} x_{ij} = 1 \qquad \text{for } i \in V_1,$$

$$\sum_{\substack{j \in V_1 \\ (i,j) \in E}} x_{ij} = 1 \qquad \text{for } j \in V_2,$$

$$\sum_{(i,j)\notin E} x_{ij} = 0,$$

$$x_{ij} \geq 0 \quad \text{for } i \in V_1,\ j \in V_2.$$

2.10. Theorem. (KUHN [10], EDMONDS [8])
(1) If (LP) is solvable then there exists a solution x^* such that

$$x^*_{ij} \in \{0,1\} \quad \textit{for all } i \in V_1,\ j \in V_2$$

and $M^* = \{(i,j) \big| x^*_{ij} = 1\}$ is an optimal matching.
(2) In this way, any optimal matching gives a solution of (LP).
(3) M^* is optimal iff variables u_i $(i \in V_1)$ and v_j $(j \in V_2)$ exist with

$$u_i + v_j \leq c_{ij} \textit{ for } (i,j) \in E,$$

$$u_i + v_j = c_{ij} \textit{ if } (i,j) \in M^*. \quad \Box$$

The variables u_i $(i \in V_1)$ and v_j $(j \in V_2)$ belong to the solution of the dual program of (LP) and are simply called the *dual variables.* Most AP-algorithms use the dual variables only to verify the computed solution according to Theorem 2.10, but in our scaling program the dual variables have the following additional meaning.

2.11. Lemma. *Let π be the dominant transversal for A defined by M^*, and let D_1 and D_2 be diagonal matrices with*

$$\begin{array}{ll} d^1_i := \exp(u_i)/\bar{a}_i & \textit{for } i = 1,...,n, \\ d^2_j := \exp(v_j) & \textit{for } j = 1,...,n. \end{array}$$

Then $\tilde{A} := P_\pi D_1 A D_2$ is an I-Matrix.

Proof: We have to show that

$$d_i^1 \cdot |a_{ij}| \cdot d_j^2 \le 1 \quad \text{and } d_{\pi(j)}^1 \cdot |a_{\pi(j),j}| \cdot d_j^2 = 1 \text{ for } i,j = 1, ..., n.$$

This is clear if $a_{ij} = 0$. Otherwise

$$\begin{array}{rcl} d_i^1 \cdot |a_{ij}| \cdot d_j^2 & = & (\bar{a}_i)^{-1} \cdot \exp(u_i) \cdot |a_{ij}| \cdot \exp(v_j) \\ & = & \exp[u_i + v_j - (\log(\bar{a}_i) - \log(|a_{ij}|)] \\ & = & \exp[u_i + v_j - c_{ij}] \le 1, \end{array}$$

since $u_i + v_j - c_{ij} \le 0$. If $i = \pi(j)$, then equality holds. □

From Lemma 2.11 we see that we can find P, D_1 and D_2 by using an AP-algorithm, and therefore Theorem 2.8 is proved.

3 Implications for partial pivoting

As discussed in Section 1, implicit partial pivoting consists in choosing in the kth elimination step some $k \ge j$ with

$$|a_{kj}^{(j-1)}| \cdot d_k^1 = \max_{i \ge j} |a_{ij}^{(j-1)}| \cdot d_i^1 \ne 0,$$

and then take $a_{kj}^{(j-1)}$ as pivot element. In our approach, the nonsingular diagonal matrix $D_1 = \mathrm{Diag}(d_1^1, ..., d_n^1)$ is chosen such that the matrix $\tilde{A} = D_1 A D_2$ is an I-matrix for some diagonal matrix D_2 (which does not affect the pivot choice). Since an I-matrix has a structure resembling an equilibrated, diagonally dominant matrix, we expect better results in Gaussian elimination than with the traditional choices. Note that if the dominant transversal is unique, the resulting scaling is invariant under prior shuffling and scaling of rows or columns, which makes it more robust than simple row equilibration.

In the special case when A is obtained by scaling and permuting a diagonally dominant matrix A_0, Theorem 2.5 implies that the pivot sequence chosen is equivalent to using the original ordering of A_0 without pivoting, and the same happens for scaled and permuted symmetric positive definite matrices. Since Gaussian elimination is stable in these cases, this nourishes the hope that we also get a good pivoting sequence in the general situation.

The experimental results obtained in Section 4 indeed suggest that this holds, but we cannot prove this, since - unlike for H-matrices and symmetric positive definite matrices - the I-matrix property is not preserved under elimination. In particular, when using Gaussian elimination *without* pivoting on an I-matrix, it can happen that that a diagonal element becomes small (or even zero) and hence is not suitable as pivot. Thus partial pivoting is still essential. This is why, for general matrices, we do not use the transversal, but rather the associated left scaling matrix with implicit partial pivoting. This combines the best of both worlds. (The perfect, but expensive way would be to restore I-dominance after each step; this would probably even allow to prove strong stability statements - but we did not consider this in detail since it is not practical.)

When using the scaling factors computed with dual variables, a disadvantage becomes apparent during triangularization when there are several elements of absolute value 1 in a column of D_1AD_2. This always happens when he dominant transversal π is not unique. In this case, often elements not from π are taken as pivots, thus prematurely destroying the good ordering. Since the permuted matrix is generally no longer an I-matrix, one such occurence may already imply that all later pivots are off the transversal, and the effectiveness of the scaling is much reduced.

Therefore we consider in the next section a method for improving the scaling factors. Our goal will be the transformation to an *I-dominant matrix* $\tilde{A}$, defined as an I-matrix with a minimal number of off-diagonal elements of absolute value 1 among all I-matrices $D_1\tilde{A}D_2$. (Note that in every I-matrix $P_\pi D_1AD_2$ there are off-diagonal elements of absolute value 1 when the dominant transversal π for A is not unique.)
The main relevance of I-dominance is that it automatically finds a good scaling, and preserves or restores good orderings in the cases where theory predicts they are good. For sufficiently bad matrices, partial pivoting must take care of the loss of I-dominance during elimination.

4 Practical Questions

For the choice of the AP-algorithm one has to consider not only running time and required work space, but also whether the program computes a dual solution. We chose a version of the LSAPR-algorithm of Burkard and Derigs [2] with starting procedure SAT3 of Carpaneto and Toth [3]. In a timing test, this combination performed best among a number of choices tested, with an (experimental) running time of the order of $O(n^{2.25})$ for full $n{\times}n$-matrices. The key concept of the LSAPR-algorithm is the so-called *shortest augmenting path method*.

Let M be a matching in a graph G, then an *M-alternating path* $P = (V(P), E(P))$ is a path in G the edges of which are alternately in M and not in M. $P_{i\to j}$ denotes an M-alternating path connecting the nodes i and j and $P_{i\to i}$ denotes an M-alternating cycle. $P_{i\to j}$ is called an *M-augmenting path* if the nodes i and j are not matched under M. Let P be an M-augmenting path (M-alternating cycle). Then

$$M \oplus P := (M \setminus E(P)) \cup (E(P) \setminus M)$$

is again a matching, and

$$l(P) := c(M \oplus P) - c(M)$$

is called the length of P. A matching M is called *extreme* if it does not allow any M-alternating cycle P with $l(P) < 0$. The SAP-method starts from any extreme matching and successively augments along shortest augmenting paths. For each of these matchings dual variables $(u, v) \in \mathbb{R}^{V_1 \cup V_2}$ are computed with

$$\begin{array}{rcll} u_i + v_j & \leq & c_{ij} & \text{for} \quad (i,j) \in E, \\ u_i + v_j & = & c_{ij} & \text{if} \quad (i,j) \in M. \end{array}$$

Using the reduced cost coefficients

$$\bar{c}_{ij} := c_{ij} - u_i - v_j > 0 \quad \text{for all edges } (i,j) \in E,$$

the shortest augmenting path can be found similar to Dijkstra's algorithm since the reduced length of an alternating path P equals the sum

$$\bar{l}(P) = \sum_{(i,j)\in E(P)\setminus M} \bar{c}_{ij}$$

with nonnegative factors $\bar{c}_{ij}$.

If the AP-program has finished with the optimal matching, the reduced costs describe the form of the I-matrix $\tilde{A} = P_\pi D_1 A D_2$ where D_1 and D_2 are defined according to Lemma 2.11. Note that

$$\begin{array}{lcl} \bar{c}_{ij} > 0 & \Leftrightarrow & d_i^1 \cdot |a_{ij}| \cdot d_j^2 < 1 \quad \text{and} \\ \bar{c}_{ij} = 0 & \Leftrightarrow & d_i^1 \cdot |a_{ij}| \cdot d_j^2 = 1. \end{array}$$

As in the elements of C, the value ∞ should be admitted for the scaling factors. Then in the program ∞ is to be replaced by a suitable large number such that values as $\infty/2$ or $\exp(\pm\infty)$ are still defined, too.

We now consider a method for improving the scaling factors to get an I-dominant matrix $\tilde{A}$, as defined in Section 3.

4.1. Lemma. *Assume that $p \in \mathbb{R}^n$ satisfies*

$$\bar{c}_{\pi(j),k} + p_j - p_k \geq 0 \quad \textit{for all } j, k = 1, ..., n.$$

Then

$$\begin{array}{l} d_{\pi(j)}^1 := \exp(-(v_j + p_j))/|a_{\pi(j),j}| \quad \textit{and} \\ d_j^2 := \exp(v_j + p_j) \end{array}$$

are scaling factors such that $\tilde{A} = P_\pi D_1 A D_2$ is an I-matrix. Moreover

$$\bar{c}_{\pi(j),k} + p_j - p_k > 0 \quad \textit{implies} \quad d_{\pi(j)}^1 |a_{\pi(j),k}| d_k^2 < 1.$$

Proof: Let $|a_{\pi(j),k}| \neq 0$, then

$$\begin{array}{rcl} d_{\pi(j)}^1 |a_{\pi(j),k}| d_k^2 & = & |a_{\pi(j),j}|^{-1} \cdot \exp(-v_j - p_j) \cdot |a_{\pi(j),k}| \cdot \exp(v_k + p_k) \\ & = & \exp(c_{\pi(j),j} - v_j - p_j + c_{\pi(j),k} + v_k + p_k) \\ & = & \exp(-(\bar{c}_{\pi(j),k} + p_j - p_k)) \leq 1. \end{array}$$

Equality holds only if $\bar{c}_{\pi(j),k} + p_j - p_k = 0$. □

Now we shall show some ways for computing $p \in \mathbb{R}^n$.

4.2. Algorithm. Equalization of the reduced costs $\bar{c}_{ij}$ in $O(n^2 \cdot k_{\max})$ operations

```
$p = 0$
do $k = 1, k_{\max}$
    do $j = 1, n$
        $y_1 = \min\{\bar{c}_{\pi(j),l} + p_j - p_l \mid l = 1, ..., n,\ l \neq j\}$
        $y_2 = \min\{\bar{c}_{\pi(l),j} + p_l - p_j \mid l = 1, ..., n,\ l \neq j\}$
        $p_j := p_j + (y_2 - y_1)/2.$
    end do
end do
```

4.3. Theorem. *Let D_1 and D_2 be diagonal matrices containing the scaling factors via equalization with $k_{\max} \geq [n/2]$ and let π be a unique dominant transversal in A. Then $\tilde{A} = P_\pi D_1 A D_2$ is an I-dominant matrix.*

For $k_{\max} = \infty$ the algorithm maximizes

$$\min\{\bar{c}_{\pi(j),k} + p_j - p_k \mid j \neq k\}$$

and thus, if π is unique for A, we can regard the scaling factors via equalization with sufficiently large $k_{\max}$ as optimal scaling factors.

Unfortunately, equalization uses much running time. Computing scaling factors computed from dual variables takes (in all our examples) significantly less time than the triangularization, but scaling via equalization with $k_{\max} = [n/2]$ costs on average 3 times as much as the factorization. So we consider an alternative.

4.4. Theorem. *(Path Selection)*
Let

$$r_{ij} := \min \bar{l}(P_{i \to j}),$$

$$p_j := \frac{1}{n} \sum_{\substack{i=1 \\ i \neq \pi(j)}}^{n} r_{ij} \quad \textit{for } i, j = 1, ..., n,\ i \neq \pi(j),$$

and let D_1 and D_2 be diagonal matrices containing the corresponding scaling factors. Then $\tilde{A} = P_\pi D_1 A D_2$ is an I-dominant matrix.

Proof: We show that

$$\bar{c}_{\pi(j),k} + p_j - p_k = 0 \text{ for } j,k = 1,...,n, \quad j \neq k$$

iff $(\pi(j),k)$ is an edge of an M-alternating cycle P of reduced length $\bar{l}(P) = 0$. We note first that for all $i,j,k = 1,...,n,\ i \neq \pi(j),\ i \neq \pi(k)$, we have

$$r_{ij} + \bar{c}_{\pi(j),k} \geq \min \bar{l}(P_{i\to k}) = r_{ik},$$
$$\bar{c}_{ik} \geq \min \bar{l}(P_{i\to k}) = r_{ik}.$$

Therefore

$$\begin{aligned}
&\bar{c}_{\pi(j),k} + p_j - p_k \\
&\quad = \bar{c}_{\pi(j),k} + \frac{1}{n}\sum_{\substack{i=1\\ i\neq\pi(j)}} r_{ij} - \frac{1}{n}\sum_{\substack{i=1\\ i\neq\pi(k)}} r_{ik} \\
&\quad = \frac{1}{n}\Big(\bar{c}_{\pi(j),k} + r_{\pi(k),j} + \bar{c}_{\pi(j),k} - r_{\pi(j),k} + \sum_{\substack{i=1\\ i\neq\pi(j),\pi(k)}} (r_{ij} + \bar{c}_{\pi(j),k} - r_{ik})\Big) \\
&\quad \geq \frac{1}{n}(\bar{c}_{\pi(j),k} + r_{\pi(k),j}).
\end{aligned}$$

Hence

$$\bar{c}_{\pi(j),k} + p_j - p_k \begin{cases} > \ 0 & \text{if } (\pi(j),k) \text{ is an edge of a zero cycle,} \\ = \ 0 & \text{otherwise.} \end{cases} \quad \square$$

Theoretically, r_{ij} can be computed by path selection of the SAP-method involving $O(n^3)$ operations, but surely a more efficient implementation can be found. However, this is not our aim, because here we only wanted to describe the possibilities to vary the scaling program. Since there are many AP-algorithms and new algorithms are still being proposed, we cannot yet decide which method is the most appropriate matching strategy and whether there is a fast implementation to optimize the scaling factors by maximizing

$$\min\{\bar{c}_{\pi(j),k} + p_j - p_k \;\big|\; j \neq k\}.$$

It may even turn out feasible to update the matching after every elimination step and taking the entry of a dominant transversal in the ith column as ith pivot.

5 Computational results

To test the partial pivoting procedure with the new scaling we used a program which modified the IMSL-routine LEQT1F for computing a triangular factorization. However, we deleted the test of singularity since, particularly for column scaled matrices, the program often returned with an error message even if good results would have been obtained (cf. Appendix). The tests were performed using single precision (27 binary digits, rounding by truncation, $\varepsilon \approx 1.5 \cdot 10^{-8}$).

LEQT1F uses an implicit 'row maximum scaling' pivoting strategy, with scaling factors

$$d_i^1 := 1/ \max_{j=1,...,n} |a_{ij}| \quad \text{for } i = 1, ..., n.$$

To obtain the 'matching scaling' strategy, we simply replaced the d_i^1 by the scaling factors obtained by our equalization procedure.

The following tables show the results for different systems $Ax = b$. We generated ten $20{\times}20$ - matrices A with randomly generated coefficients from the range

$$[-1, +1] \cdot 10^{[-8,+8]}$$

(with uniformly distributed factor and exponent) and for each A ten different vectors b. As a measure of accuracy we used the relative number of decimals the residual was reduced,

$$d(A, b) := -\log_{10} \left(\max_{i=1,...,n} \frac{|(Ax-b)_i|}{(|A|\ |x| + |b|)_i} \right),$$

which is invariant under row and column scaling.

Table 1 displays the minimal value, the maximal value and the average of $d(A, b)$ and of the number of decimals gained, i.e., the difference $d_{matching}(A, b) - d_{rowsum}(A, b)$. Moreover we show in which column M the first element not from π was taken as the pivot, i.e., in which column the pivots left the sequence designed by the matching strategy.

Table 1. Residual accuracy for 100 full 20×20 - systems

	row maximum	matching	decimals gained	M
Minimum	4.39	6.53	-.48	11
Average	6.63	7.47	0.85	17
Maximum	7.78	7.80	3.29	20
Time(msec)	32.1	135.0		

Mainly in those cases where the pivot selection does not deviate from the matching for a long time, high accuracy can be obtained. In limited tests of matrices with many zeros the results are surprising. Here we used the same matrices as above, but we retained in each row an average of 6 coefficients out of the 20 and replaced the other coefficients by zero. Now the matching strategy can even obtain an average of higher accuracy than in the case of full matrices, while the results of the row maximum strategy are very unsatisfactory. An extra advantage of the matching strategy in the sparse case turned out to be that the matrices L and U were often more sparse than with the row maximum strategy.

Table 2. Residual accuracy for 100 sparse 20×20 - systems
(calculated with full matrix code)

	row maximum	matching	decimals gained	M
Minimum	0.01	5.69	-.27	19
Average	4.07	7.60	3.53	20
Maximum	7.92	7.98	7.56	20
Time(msec)	31.7	122.0		

6 Conclusion

In this paper we presented the basic theory and some simple examples showing the potential of a new scaling method for Gaussian elimination with partial pivoting, based on solving an associated assignment problem. The method is capable of restoring diagonally dominant matrices from scrambled and scaled versions of it, and it gives good scalings in the general case. However, solving the assignment problem is in itself a nontrivial task, and for applications to larger problems, further work must be done.

The timing results for the sparse matrices are of course not realistic since we used a full matrix implementation. While unweighted sparse assignment problems can be solved very fast, we do not know how fast the weighted sparse assignment problem can be solved, in particular since we need the dual solution as well. For some algorithms for solving sparse assignment problems see Derigs & Metz [7].

Another interesting possibility in the large scale case is the use of an incomplete factorization (without pivoting) of the I-matrix of Theorem 2.8, as a preconditioner for iterative methods for nonsymmetric linear systems like QMR (Freund & Nachtigal [9]). In this case, equalization considerations play no longer a role, which speeds things up. Moreover, if a dominant matching cannot be computed fast enough, one can probably design fast heuristics for finding a nearly dominant matching, which might suffice since the coefficients change anyway after each eliminiation step.

However, a proper treatment of these matters is beyond the scope of the present paper.

Appendix: Some remarks on published implementations of Gaussian elimination

1. As remarked above, $LEQT1F$ is sometimes far too cautious in assessing nonsingularity. Let the factorization routine $LEQT1F$ take $a_{ij}^{(j-1)}$ as pivot in the jth step of reduction and let

$$\omega := |a_{ij}^{(j-1)}| \cdot d_i^1.$$

$LEQT1F$ gives failure if $\omega + n = n$, and the program returns with an error message. This is appropriate when all elements of A are of the same order unity, and the routine tries to achieve this by standard row equilibration. But this is not always good enough; for example $LEQT1F$ stops with $\omega = 10^{-10}$ in the first step of reduction of the nonsingular matrix

$$\begin{pmatrix} 10^{-5} & 10^{+5} \\ 0 & 10^{+5} \end{pmatrix}$$

(which is already the upper triangular matrix U). We suggest just to give a warning if the test reports singularity and then to continue the reduction.

2. We also tested our scaling method on $LINGL$, a triangular factorization procedure developed by Dekker [6]. Surprisingly, the resulting accuracy was often more than one decimal better. After inspection, the crucial difference to $LEQT1F$ turned out to be the accumulation of the inner products after selecting the pivot and exchanging the rows.

The accumulation of

$$A_{ii} - \sum L_{ij} \cdot R_{ji} \tag{5}$$

is done in $LEQT1F$ by repeated subtraction and in $LINGL$ according to

$$A_{ii} - \left(\sum L_{ij} \cdot R_{ji}\right). \tag{6}$$

By Wilkinson [14], Ch. I, (32.9/10), the rounding error in the computation of $a_1b_1 + ... + a_nb_n$ by repeated addition is

$$(a_1b_1\varepsilon_1 + ... + a_nb_n\varepsilon_n)(1+\varepsilon)$$

where

$$|\varepsilon| \le 2^{-t}, \quad |\varepsilon_1| < \frac{3}{2}n \cdot 2^{-2t_2}, \quad |\varepsilon_r| < \frac{3}{2}(n+2-r) \cdot 2^{-2t_2},$$

with t, t_2 depending on the accuracy of the computer. This shows that permuting a term of the summation from first position to last reduces its worst case contribution to the total error by a factor $n/2$. Since in an I-matrix the diagonal terms are the largest terms in their rows, the pivots A_{ii} tend to be the largest element in the sum (5) and (6). This explains why the second form (6) gives more accurate results.

In the mean time, IMSL changed the factorization routine, thus eliminating the weaknesses mentioned.

References

1. R. C. Bose; B. Manvel. Introduction to Combinatorial Theory. Wiley, Colorado State Univ. 1983.

2. R. E. Burkard; U. Derigs. Assignment and Matching Problems: Solution Methods with FORTRAN-Programs. Lecture Notes in Econ. and Math. Systems, Springer, Berlin 1980.

3. G. Carpaneto; P. Toth. Solution of the assignment problem (Algorithm 548). ACM Trans. Math. Softw. (1980), 104-111.

4. S. D. Conte and C. de Boor. Elementary Numerical Analysis. 3rd ed., McGraw-Hill, Auckland 1981.

5. A. R. Curtis; J. K. Reid. On the automatic scaling of matrices for Gaussian elimination. J. Inst. Math. Appl. 10 (1972), 118-124.

6. T. J. Dekker. ALGOL 60 Procedures in Numerical Algebra. Mathematical Center Tracts 22 (1968), 20-21.

7. U. Derigs; A. Metz. An efficient labeling technique for solving sparse assignment problems. Computing 36 (1986), 301-311.

8. J. Edmonds. Maximum matching and a polyhedron with 0-1-vertices. J. Res. Nat. Bur. Stand. B69 (1965), 125-130.

9. R.W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitean linear systems. Numer. Math. 60 (1991), 315-339.

10. H. W. Kuhn. The Hungarian method for the assignment problem. Naval Research Logistics Quarterly 2 (1955) 83.

11. B. N. Parlett; T. L. Landis. Methods for scaling to double stochastic form. Linear Algebra Appl. 48 (1982), 53-79.

12. J. R. Rice. Matrix Computation and Mathematical Software. McGraw-Hill, New York 1981.

13. J. H. Wilkinson. Error Analysis of Direct Methods of Matrix Inversion. J. Assoc. Comput. Mach. 8 (1961).

14. J. H. Wilkinson. Rounding Errors in Algebraic Processes. Prentice Hall, Englewood Cliffs, N.J. 1963.