# A SYMMETRY PRESERVING ALGORITHM FOR MATRIX SCALING*

PHILIP A. KNIGHT†, DANIEL RUIZ‡, AND BORA UÇAR§

**Abstract.** We present an iterative algorithm which asymptotically scales the $\infty$-norm of each row and each column of a matrix to one. This scaling algorithm preserves symmetry of the original matrix and shows fast linear convergence with an asymptotic rate of 1/2. We discuss extensions of the algorithm to the one-norm, and by inference to other norms. For the 1-norm case, we show again that convergence is linear, with the rate dependent on the spectrum of the scaled matrix. We demonstrate experimentally that the scaling algorithm improves the conditioning of the matrix and that it helps direct solvers by reducing the need for pivoting. In particular, for symmetric matrices the theoretical and experimental results highlight the potential of the proposed algorithm over existing alternatives.

**Key words.** Sparse matrices, matrix scaling, equilibration

**AMS subject classifications.** 05C50, 65F35, 65F50

**1. Introduction.** Scaling a matrix consists of pre- and post-multiplying the original matrix by two diagonal matrices. We consider the following scaling problem: given a large, sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, find two positive diagonal matrices $\mathbf{D}$ and $\mathbf{E}$ such that all rows and columns of the scaled matrix $\widehat{\mathbf{A}} = \mathbf{DAE}$ have the same length in the $\infty$-norm. We propose an iterative algorithm for this purpose which preserves symmetry (when present), and we investigate its convergence properties. We discuss the extension of the algorithm to the 1-norm, and describe how it can be used for other norms as well.

Scaling or equilibration of data in linear systems of the form $\mathbf{Ax} = \mathbf{b}$ can provide significant computational benefits. In this setting, we form a scaled matrix $\widehat{\mathbf{A}} = \mathbf{DAE}$ and solve the equation $\widehat{\mathbf{A}}\hat{\mathbf{x}} = \hat{\mathbf{b}}$, where $\hat{\mathbf{x}} = \mathbf{E}^{-1}\mathbf{x}$ and $\hat{\mathbf{b}} = \mathbf{Db}$, with the expectation that the scaled system has better conditioning and hence provides more accurate results. Scaling can also be helpful in designing algorithms for linear systems. For example, Bunch and Parlett [**?**] prove the stability of their factorization method by assuming that the rows (hence the columns) of the symmetric input matrix have $\infty$-norm equal to 1.

There are several well-known algorithms (see for example [**?**, Section 4.12], [**?**, Section 3.5.2], and [**?**]) for scaling matrices. The row and column scaling methods are amongst the simplest. The row scaling method divides each row in the original matrix by its norm. Different norms, such as the $\infty$-norm or the 1-norm, may be considered, depending on the application. Column scaling is equivalent to row scaling on $\mathbf{A}^T$. A more elaborate scaling method is proposed by Curtis and Reid [**?**] and is implemented as `MC29` in the HSL Mathematical Software Library. This method aims to make the nonzeros of the scaled matrix close to one by minimizing the sum of the squares of the

*This work was supported in part by "Agence Nationale de la Recherche" through SOLSTICE project (ANR-06-CIS6-010).

†Department of Mathematics and Statistics, University of Strathclyde, Glasgow G1 1XH, Scotland (`p.a.knight@strath.ac.uk`).

‡ENSEEIHT-IRIT, 2 rue Camichel, 31071, Toulouse, France (`Daniel.Ruiz@enseeiht.fr`). The work of this author was supported in part by the Rutherford Appleton Laboratory, Computational Science and Engineering Department, Chilton, Didcot, Oxfordshire, UK.

§Centre National de la Recherche Scientifique, LIP (CNRS - ENS Lyon - INRIA - UCBL - Université de Lyon), 46, allée d'Italie, ENS Lyon, F-69364, France (`bora.ucar@ens-lyon.fr`).

logarithms of the moduli of the nonzeros. MC29 reduces that sum in a global sense and can provide benefits over a wide range of sparse matrices. The routine MC30 in the HSL library is a variant of MC29 for symmetric matrices. Scaling can also be combined with permutations (see [**?**] and the HSL routine MC64). A matrix is first permuted to maximise the product of the absolute values of the entries on the diagonal. Then the matrix is scaled so that the diagonal entries equal one and the off-diagonals are less than or equal to one. This has been shown [**?**] to be useful for finding a good sequence of pivots for sparse direct solvers and for building good incomplete LU preconditioners for iterative methods.

In the 1960s, optimality properties were established in terms of condition numbers for scaled matrices all of whose rows or columns have norm one [**?**, **?**, **?**]. In particular, the optimal scaling of a matrix $\mathbf{A}$ which minimizes the condition number in the $\infty$-norm is characterized by both $\mathbf{DAE}$ and $\mathbf{E}^{-1}\mathbf{A}^{-1}\mathbf{D}^{-1}$ having equal row sums of absolute values [**?**]. Other optimality results for one sided scaling, i.e., $\mathbf{DA}$ or $\mathbf{AE}$ are also shown [**?**], again based on the equivalence of the norms of rows or columns in the $\infty$- and 1-norms.

The paper is organized as follows. In Section 2 we introduce the proposed algorithm, derive a convergence analysis, and show some particular properties it yields for scaled matrices. In Section 3, we consider the algorithm in norms other than the $\infty$-norm. Following a discussion in [**?**], we establish precise conditions for convergence in the 1-norm, and we look at the rate of convergence, too. We present numerical results in Section 4 and finish with some concluding remarks in Section 5. While proving the convergence of the proposed algorithm in 1-norm, we deviate somewhat from the approach in [**?**]. An appendix includes a detailed explanation of the differences.

**2. The algorithm.** Consider a general $m \times n$ real matrix $\mathbf{A}$. For $i = 1, \ldots, m$, let $\mathbf{r}_i = \mathbf{a}_{i*}^T \in \mathbb{R}^{n\times 1}$ denote the $i$th row of $\mathbf{A}$ as a vector, and for $j = 1, \ldots, n$, let $\mathbf{c}_j = \mathbf{a}_{*j} \in \mathbb{R}^{n\times 1}$ denote the $j$th column of $\mathbf{A}$. Let $\mathbf{R}$ and $\mathbf{C}$ denote the $m \times m$ and $n \times n$ diagonal matrices given by:

$$\mathbf{R} = \operatorname{diag}\left(\sqrt{\|\mathbf{r}_i\|_\infty}\right)_{i=1,\ldots,m} \quad \text{and} \quad \mathbf{C} = \operatorname{diag}\left(\sqrt{\|\mathbf{c}_j\|_\infty}\right)_{j=1,\ldots,n} \tag{2.1}$$

where $\|\cdot\|_\infty$ stands for the $\infty$-norm of a real vector (that is the maximum entry in absolute value; sometimes called the max norm). If a row (or a column) in $\mathbf{A}$ has all entries equal to zero, we replace the diagonal entry in $\mathbf{R}$ (or $\mathbf{C}$ respectively) by 1. In the following, we will assume that this does not happen.

One can scale the matrix $\mathbf{A}$ on both sides, forming the scaled matrix $\widehat{\mathbf{A}}$ as

$$\widehat{\mathbf{A}} = \mathbf{R}^{-1}\mathbf{A}\mathbf{C}^{-1} \,. \tag{2.2}$$

The idea of the algorithm we propose is to iterate on that process, resulting in Algorithm 1. The algorithm converges when

$$\max_{1\le i\le m}\left\{\left|1 - \|\mathbf{r}_i^{(k)}\|_\infty\right|\right\} \;\le\; \varepsilon \quad \text{and} \quad \max_{1\le j\le n}\left\{\left|1 - \|\mathbf{c}_j^{(k)}\|_\infty\right|\right\} \;\le\; \varepsilon \tag{2.3}$$

for a given value of error tolerance parameter $\varepsilon > 0$.

**2.1. A salient property.** We highlight that the proposed iterative scaling procedure preserves the symmetry in the original matrix. In fact, this is one of our main motivations. If the given matrix $\mathbf{A}$ is symmetric, then the diagonal matrices $\mathbf{R}$ and

**Algorithm 1** Simultaneous row and column scaling in the $\infty$-norm

$\mathbf{A}^{(0)} \leftarrow \mathbf{A}$
$\mathbf{D}^{(0)} \leftarrow \mathbf{I}_m$
$\mathbf{E}^{(0)} \leftarrow \mathbf{I}_n$
**for** $k = 0, 1, 2, \ldots$ **until** convergence **do**
  $\mathbf{R} \leftarrow \operatorname{diag}\left(\sqrt{\|\mathbf{r}_i^{(k)}\|_\infty}\right)_{i=1,\ldots,m}$ ▶ $\mathbf{r}_i^{(k)}$ is the $i$th row of $\mathbf{A}^{(k)}$
  $\mathbf{C} \leftarrow \operatorname{diag}\left(\sqrt{\|\mathbf{c}_j^{(k)}\|_\infty}\right)_{j=1,\ldots,n}$ ▶ $\mathbf{c}_j^{(k)}$ is the $j$th column of $\mathbf{A}^{(k)}$
  $\mathbf{A}^{(k+1)} \leftarrow \mathbf{R}^{-1}\mathbf{A}^{(k)}\mathbf{C}^{-1}$
  $\mathbf{D}^{(k+1)} \leftarrow \mathbf{D}^{(k)}\,\mathbf{R}^{-1}$
  $\mathbf{E}^{(k+1)} \leftarrow \mathbf{E}^{(k)}\,\mathbf{C}^{-1}$

$\mathbf{C}$ in (2.1) are equal and, consequently, matrix $\widehat{\mathbf{A}}$ in (2.2) is symmetric, as is the case for the matrices $\mathbf{A}^{(k)}$ at any iteration in Algorithm 1. This is not the case for most scaling algorithms which alternately scale rows followed by columns or vice-versa.

In the case of unsymmetric matrices, one may consider the use of the Sinkhorn–Knopp iterations [**?**] with the $\infty$-norm in place of the 1-norm. This method simply normalizes all rows and then all columns of $\mathbf{A}$, and iterates on this process until convergence. In $\infty$-norm, the convergence is achieved after one single step. Because of its simplicity, this method is very appealing. There are however differences. The Sinkhorn–Knopp iteration may provide different results when applied to $\mathbf{A}^T$. On the other hand Algorithm 1 provides exactly the same results when applied to $\mathbf{A}$ or $\mathbf{A}^T$ in the sense that the scaled matrix obtained on $\mathbf{A}^T$ is the transpose of that obtained on $\mathbf{A}$. We have quoted the Sinkhorn–Knopp method in particular because it has been originally proposed to obtain *doubly stochastic* matrices (that is nonnegative matrices with all rows and columns of 1-norm equal to one), and we shall come back to this issue with respect to Algorithm 1 in Section 3.

**2.2. Convergence rate.** The proposed algorithm convergences quickly to a matrix whose rows and columns have $\infty$-norm equal to one as stated below.

THEOREM 2.1. *If $\mathbf{A}$ is an $m \times n$ matrix with no zero rows or columns, then Algorithm 1 produces a sequence of matrices $\mathbf{A}^{(0)}, \mathbf{A}^{(1)}, \ldots$ which converge linearly to a matrix all of whose rows and columns have infinity norm equal to one. The asymptotic rate of convergence is 1/2.*

*Proof.* After the first iteration of the algorithm, all the entries in $\mathbf{A}^{(1)}$ are less than or equal to one in absolute value. This is because all entries $a_{ij}$ in $\mathbf{A}$ are divided by the square roots of two numbers, $\|\mathbf{r}_i^{(k)}\|_\infty$ and $\|\mathbf{c}_j^{(k)}\|_\infty$ respectively, each one of them being greater than or equal to $|a_{ij}|$ itself.

For any subsequent iteration ($k \geq 1$), consider the $\infty$-norm of any row $\mathbf{r}_i^{(k)}$ or column $\mathbf{c}_j^{(k)}$, and let indices $\ell$ and $p$ satisfy the equalities $|a_{ip}^{(k)}| = \|\mathbf{r}_i^{(k)}\|_\infty$ and $|a_{\ell j}^{(k)}| = \|\mathbf{c}_j^{(k)}\|_\infty$. With these notations, we can easily verify that both entries $a_{ip}^{(k+1)}$ and $a_{\ell j}^{(k+1)}$ in the scaled matrix $\mathbf{A}^{(k+1)}$ are greater, in absolute value, than the square root of the corresponding value at iteration $k$, and are still less than one. We can write

$$1 \geq |a_{ip}^{(k+1)}| = \frac{|a_{ip}^{(k)}|}{\sqrt{\|\mathbf{r}_i^{(k)}\|_\infty}\sqrt{\|\mathbf{c}_p^{(k)}\|_\infty}} = \frac{\sqrt{|a_{ip}^{(k)}|}}{\sqrt{\|\mathbf{c}_p^{(k)}\|_\infty}} \geq \sqrt{|a_{ip}^{(k)}|}$$

since $|a_{ip}^{(k)}| = \|\mathbf{r}_i^{(k)}\|_\infty$ and $\|\mathbf{c}_p^{(k)}\|_\infty \leq 1$ for any $k \geq 1$. Similarly,

$$\sqrt{|a_{\ell j}^{(k)}|} \leq |a_{\ell j}^{(k+1)}| \leq 1 \ ,$$

for any $k \geq 1$. From this, we can write that the iterations in Algorithm 1 provide scaled matrices $\mathbf{A}^{(k)}$, $k = 1, 2, \ldots$ with the following properties

$$\text{for all } k \geq 1,\ 1 \leq i \leq m,\ \ \sqrt{\|\mathbf{r}_i^{(k)}\|_\infty} \leq |a_{ip}^{(k+1)}| \leq \|\mathbf{r}_i^{(k+1)}\|_\infty \leq 1 \ , \tag{2.4}$$

and

$$\text{for all } k \geq 1,\ 1 \leq j \leq n,\ \ \sqrt{\|\mathbf{c}_j^{(k)}\|_\infty} \leq |a_{\ell j}^{(k+1)}| \leq \|\mathbf{c}_j^{(k+1)}\|_\infty \leq 1 \ , \tag{2.5}$$

which shows that both row and column norms must converge to 1. To conclude our demonstration, we just need to see that for all $i$

$$1 - \|\mathbf{r}_i^{(k+1)}\|_\infty = \frac{1 - \|\mathbf{r}_i^{(k+1)}\|_\infty^2}{1 + \|\mathbf{r}_i^{(k+1)}\|_\infty} \leq \frac{1 - \|\mathbf{r}_i^{(k)}\|_\infty}{1 + \|\mathbf{r}_i^{(k+1)}\|_\infty},$$

and that similar equations hold for the columns as well. □

A small example [**?**] shows that this asymptotic rate is sharp. Let us consider the following $2 \times 2$ matrix with a badly scaled row

$$\mathbf{A} = \begin{pmatrix} \alpha & \alpha \\ 1 & 1 \end{pmatrix}.$$

If $\alpha \ll 1$, then iteration $k$ ($k \geq 1$) of the algorithm provides the following matrices:

$$\mathbf{D}^{(k)} = \begin{pmatrix} \alpha^{-(1-\frac{1}{2^k})} & 0 \\ 0 & 1 \end{pmatrix},\ \mathbf{A}^{(k)} = \begin{pmatrix} \alpha^{\frac{1}{2^k}} & \alpha^{\frac{1}{2^k}} \\ 1 & 1 \end{pmatrix},\ \mathbf{E}^{(k)} = \mathbf{I}_2 \ .$$

When the algorithm converges, the scaled matrix $\widehat{\mathbf{A}}$ is the matrix with all ones, $\mathbf{D}$ has its first diagonal entry equal to $\alpha^{-1}$, and $\mathbf{E}$ stays as the identity matrix. In this case, the linear rate of $\frac{1}{2}$ is met.

**2.3. Comparison with Bunch's algorithm.** As we have stated before, Algorithm 1 is well suited for symmetric scaling of symmetric matrices. For the $\infty$-norm case, Bunch [**?**] also developed an efficient symmetric scaling algorithm. We highlight some differences between the two algorithms. Bunch's algorithm processes the rows of the lower triangular part of the given matrix in such a way that the largest entry seen in each row is made to be $\pm 1$ in the partially scaled matrix. At the end, the scaling found for the $i$th row is applied to the $i$th column of the whole matrix. Bunch's algorithm runs in $\mathcal{O}(\text{nnz})$-time, equivalent to one iteration of the proposed algorithm. Since the scaling value of a row is calculated using the scaled values of its nonzero entries in the lower triangular part, there is a dependency between the computation of the scaling values. The dependencies can be an obstacle to reduce the running time in a parallel computing environment. For example, Bunch's algorithm has to run sequentially to scale a tridiagonal matrix. On the other hand, Algorithm 1 is more amenable to parallelism (as was shown before [**?**]). One may consider permuting a matrix to take advantage of parallelism. But care should be taken, as Bunch's algorithm is sensitive to symmetric permutations applied to the original matrix (any diagonal

nonzero can be symmetrically permuted to the first position so that that entry is one in the scaled matrix). On the other hand, the proposed algorithm is independent of any permutations (not even necessarily symmetric) applied to the original matrix in the sense that the scaling matrices computed for the permuted matrix would be equal to the permuted scaling matrices computed for the original matrix.

**3. Extensions to other norms.** A natural idea would be to change the norm used in Algorithm 1, and to try the 2-norm or the 1-norm because of the optimal properties they induce (see [**?**, **?**]), and still expect convergence (where all rows and columns are of length 1 in the corresponding norm). We shall see, in the remainder of this section, that this will usually, but not always, work and we investigate the potential and limitations of such extensions. Compared to the $\infty$-norm, where different algorithms can raise very different solutions, scalings in the 1-norm or the 2-norm, when they exist, can be considered as unique as the scaled matrix is essentially unique.

With respect to the extension of Algorithm 1 to the scaling of matrices in other norms, the case of the 1-norm is central. Indeed, Rothblum et al. [**?**, page 13] showed that the problem of scaling a matrix $\mathbf{A}$ in the $\ell_p$-norm, for $1 < p < \infty$, can be reduced to the problem of scaling the $p$th *Hadamard power of* $\mathbf{A}$, i.e., the matrix $\mathbf{A}^{[p]} = [a_{ij}^p]$, in the 1-norm. We can apply that discussion to Algorithm 1 by replacing the matrix $\mathbf{A}$ with $\mathbf{A}^{[p]}$, and then by taking the *Hadamard* $p$th root, e.g., $\mathbf{D}_1^{[1/p]} = [d_{ii}^{1/p}]$, of the resulting iterates. For this reason, we shall analyse the convergence properties of Algorithm 1 for the 1-norm only, knowing that these will implicitly drive the conclusions for any of the $\ell_p$ norms, for $1 < p < \infty$.

**3.1. Background.** The idea of equilibrating a matrix such that the 1-norm of the rows and columns are all 1 is not new, dating back to at least the 1930s (see some historical remarks by Knight [**?**]). Here, we briefly review some of the previous work.

Sinkhorn and Knopp [**?**] studied a method for scaling square nonnegative matrices to doubly stochastic form, that is a nonnegative matrix with all rows and columns of equal 1-norm. Sinkhorn [**?**] originally showed that: *Any positive square matrix of order $n$ is diagonally equivalent to a unique doubly stochastic matrix of order $n$, and the diagonal matrices which take part in the equivalence are unique up to scalar factors.* Later, a different proof for the existence part of Sinkhorn's theorem with some elementary geometric interpretations was given [**?**].

Sinkhorn's result was further extended to the case of nonnegative, nonzero matrices [**?**]. A few definitions are necessary to state the result. A square $n \times n$ nonnegative matrix $\mathbf{A} \geq 0$ is said to have *support* if there exists a permutation $\sigma$ such that $a_{i,\sigma(i)} > 0$, for $1 \leq i \leq n$. Note that matrices not having support are matrices for which no full transversal can be found (see [**?**, page 107]), i.e., there is no column permutation making the diagonal zero-free, and are thus *structurally singular*. A matrix $\mathbf{A}$ is said to have *total support* if every positive entry in $\mathbf{A}$ can be permuted into a positive diagonal with a column permutation. A nonnegative nonzero square matrix $\mathbf{A}$ of size $n > 1$ is said to be *fully indecomposable* if there does not exist permutation matrices $\mathbf{P}$ and $\mathbf{Q}$ such that $\mathbf{PAQ}$ is of the form

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{0} & \mathbf{A}_{22} \end{pmatrix}, \tag{3.1}$$

with $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ being square matrices. The term bi-irreducible is also used for fully indecomposable matrices [**?**, Chapter 6]. Sinkhorn and Knopp [**?**] established that their scaling algorithm, which simply iterates on normalizing all rows and columns

in the matrix $\mathbf{A}$ alternately, converges to a doubly stochastic limit $\widehat{\mathbf{A}}$ if and only if the matrix $\mathbf{A}$ has support. The doubly stochastic limit $\widehat{\mathbf{A}}$ can be represented as $\mathbf{DAE}$ (meaning that $\mathbf{A}$ and $\widehat{\mathbf{A}}$ are diagonally equivalent) if and only if $\mathbf{A}$ has total support and, if the support of $\mathbf{A}$ is not total, then there must be a positive entry in $\widehat{\mathbf{A}}$ converging to 0. Additionally, it has been shown that the diagonal matrices $\mathbf{D}$ and $\mathbf{E}$, when they exist, are unique up to some scalar factors if and only if the matrix $\mathbf{A}$ is fully indecomposable [**?**]. Brualdi et al. [**?**] independently showed the same diagonal equivalence between $\mathbf{A}$ and a doubly stochastic matrix, when $\mathbf{A}$ is a direct sum of fully indecomposable matrices. It has been shown that a matrix $\mathbf{A}$ has total support if and only if there exist permutation matrices $\mathbf{P}$ and $\mathbf{Q}$ such that $\mathbf{PAQ}$ is a direct sum of fully indecomposable matrices [**?**, Theorem 1 (ii)].

Different contributions have also been made in the study of convergence of the Sinkhorn–Knopp method under various hypothesis. Geometric convergence rate for positive [**?**] and nonnegative matrices with total support [**?**] have been shown. The converse of the second result has also been established [**?**], i.e., geometric convergence of the Sinkhorn–Knopp algorithm implies total support for a nonnegative matrix. The explicit rates of convergence for fully indecomposable matrices are given by Knight [**?**].

Parlett and Landis [**?**] present three iterative scaling algorithms with experimental evidence of better average and worst-case convergence behavior than the Sinkhorn–Knopp method (in at least one of the three algorithms). They also give a generalized version of the convergence theorem of Sinkhorn and Knopp [**?**], including a characterization of scaling algorithms that will converge to a doubly stochastic matrix when the starting matrix $\mathbf{A}$ has support. Such algorithms are called *diagonal product increasing* (DPI) algorithms. In Appendix, we recall and partially extend Parlett and Landis's results [**?**, Theorem 1]. These extensions are needed to specify generic properties that our algorithm fulfills which are sufficient to ensure convergence of scaling algorithms in the 1-norm in general.

In the remainder of this section, we first establish that Algorithm 1 converges in the 1-norm as well as in any $p$-norm, for any $p < \infty$, so long as $\mathbf{A}$ satisfies appropriate conditions. We also analyze the convergence rate of the algorithm in the 1-norm (separately for symmetric and unsymmetric matrices), allowing a fuller comparison with other balancing algorithms.

**3.2. Convergence analysis.** Algorithm 1 produces a sequence of matrices diagonally equivalent to the starting matrix $\mathbf{A} = \mathbf{A}^{(0)}$ using the iteration

$$\begin{aligned} \mathbf{A}^{(k)} &= \left(a_{ij}^{(k)}\right) = \mathbf{D}^{(k)}\mathbf{A}\mathbf{E}^{(k)}, \quad k = 1, 2, \ldots, \\ \mathbf{D}^{(k)} &= \operatorname{diag}\left(d_1^{(k)}, \ldots, d_n^{(k)}\right), \\ \mathbf{E}^{(k)} &= \operatorname{diag}\left(e_1^{(k)}, \ldots, e_n^{(k)}\right), \end{aligned} \tag{3.2}$$

where $\mathbf{D}^{(0)} = \mathbf{E}^{(0)} = \mathbf{I}$. For convenience, let $r_i^{(k)}$, $i = 1, \ldots, n$, and $c_j^{(k)}$, $j = 1, \ldots, n$, denote the 1-norm of rows and columns respectively:

$$\begin{aligned} r_i^{(k)} &= \|\mathbf{r}_i^{(k)}\|_1 = \sum_{j=1}^{n} |a_{ij}^{(k)}|, \\ c_j^{(k)} &= \|\mathbf{c}_j^{(k)}\|_1 = \sum_{i=1}^{n} |a_{ij}^{(k)}|, \end{aligned} \tag{3.3}$$

so that $d_i^{(k)} = 1/\sqrt{r_i^{(k)}}$ and $e_j^{(k)} = 1/\sqrt{c_j^{(k)}}$. Since $\mathbf{A}$ and $|\mathbf{A}|$ scale in the same way we can assume that $\mathbf{A} \geq 0$. Under this simplification, the 1-norms of the rows and columns reduce to the row and column sums respectively, $r_i^{(k)} = \sum_{j=1}^n a_{ij}^{(k)}$ and $c_j^{(k)} = \sum_{i=1}^n a_{ij}^{(k)}$, and to generalize our results to any matrix, one just needs to extend the definition of a doubly stochastic matrix so that the absolute value of the matrix under consideration is doubly stochastic.

THEOREM 3.1. *Given the sequence* (3.2) *of diagonal equivalents for* $\mathbf{A}$*, in which*

$$a_{ij}^{(k+1)} = \frac{a_{ij}^{(k)}}{\sqrt{r_i^{(k)}}\sqrt{c_j^{(k)}}}, \quad 1 \leq i, j \leq n,$$

*with* $r_i^{(k)}$ *and* $c_j^{(k)}$ *given by* (3.3)*:*

1. *If* $\mathbf{A}$ *has support, then* $\mathbf{S} = \lim_{k\to\infty} \mathbf{A}^{(k)}$ *exists and is doubly stochastic.*
2. *If* $\mathbf{A}$ *has total support, then both* $\mathbf{D} = \lim_{k\to\infty} \mathbf{D}^{(k)}$ *and* $\mathbf{E} = \lim_{k\to\infty} \mathbf{E}^{(k)}$ *exist and* $\mathbf{S} = \mathbf{DAE}$*.*

We will prove this theorem in a more general setting, following an approach by Parlett and Landis [**?**], by establishing that the results in Theorem 3.1 hold for any iterative scaling algorithm having the following properties.

(P1*) both sequences of rows and column scaling factors products

$$\prod_{i=1}^n d_i^{(k)} \quad \text{and} \quad \prod_{i=1}^n e_i^{(k)}, \ k = 1, 2, \ldots$$

are monotonically increasing;

(P2) the 1-norm of rows and columns respectively converge to 1, i.e.,

$$\lim_{k\to\infty} r_i^{(k)} = 1 \quad \text{and} \quad \lim_{k\to\infty} c_j^{(k)} = 1, \quad 1 \leq i, j \leq n\,. \tag{3.4}$$

The full consequences of these properties, not least that they imply convergence of DPI algorithms, are detailed in the Appendix. We have made minor changes to the properties used by Parlett and Landis [**?**] so as to include Algorithm 1 in the analysis. In summary, our property (P1*) is more restrictive than that of Parlett and Landis who require the *product* of the products to be monotonically increasing. Property (P2) relaxes the two other hypothesis made by Parlett and Landis when defining DPI algorithms, but is sufficient, combined with (P1*), to obtain the conclusions in Theorem 3.1.

Before the proof, we note the following observations. By the arithmetic-geometric mean inequality, we have

$$\prod_{i=1}^n r_i^{(k+1)} \leq \left(\frac{1}{n}\sum_{i=1}^n r_i^{(k+1)}\right)^n = \left(\frac{1}{n}\sum_{1\leq i,j\leq n} \frac{a_{ij}^{(k)}}{\sqrt{r_i^{(k)}}\sqrt{c_j^{(k)}}}\right)^n$$

for all $k$, with the same inequality for $\prod_{j=1}^n c_j^{(k+1)}$ since the sum of the column sums must equal the sum of the row sums. Additionally, using the Cauchy-Schwarz inequality

on the vectors $\left(\sqrt{a_{ij}^{(k)}}/\sqrt{r_i^{(k)}}\right)_{1\le i,j\le n}$ and $\left(\sqrt{a_{ij}^{(k)}}/\sqrt{c_j^{(k)}}\right)_{1\le i,j\le n}$,

$$\sum_{1\le i,j\le n}\frac{a_{ij}^{(k)}}{\sqrt{r_i^{(k)}}\sqrt{c_j^{(k)}}}\le\sqrt{\sum_{1\le i,j\le n}\frac{a_{ij}^{(k)}}{r_i^{(k)}}}\sqrt{\sum_{1\le i,j\le n}\frac{a_{ij}^{(k)}}{c_j^{(k)}}}=\sqrt{n}\sqrt{n}\,,$$

and thus, for all $k\ge 0$, we have

$$\prod_{i=1}^{n}r_i^{(k+1)}\le\left(\frac{1}{n}\sum_{i=1}^{n}r_i^{(k+1)}\right)^n\le 1\ \text{ and }\ \prod_{j=1}^{n}c_j^{(k+1)}\le\left(\frac{1}{n}\sum_{j=1}^{n}c_j^{(k+1)}\right)^n\le 1\,. \quad (3.5)$$

Notice that after the first iteration all the entries in matrix $\mathbf{A}^{(k)}$ are less than or equal to 1, since both $r_i^{(k)}$ and $c_j^{(k)}$ are greater than $a_{ij}^{(k)}$ by construction.

*Proof.* [Theorem 3.1] By (3.5) and by the fact that the iterates in Algorithm 1 satisfy

$$\frac{\prod_{i=1}^{n}d_i^{(k+1)}}{\prod_{i=1}^{n}d_i^{(k)}}=\frac{1}{\sqrt{\prod_{i=1}^{n}r_i^{(k)}}}$$

(and similarly for the column scaling factors) we see that property (P1*) holds.

Establishing (P2) is not quite so straightforward. From the arithmetic-geometric mean inequality and (3.5),

$$\prod_{i=1}^{n}r_i^{(k)}c_i^{(k)}\le\left\{\sum_{i=1}^{n}\frac{1}{2n}\left(r_i^{(k)}+c_i^{(k)}\right)\right\}^{2n}\le 1\,.$$

From (P1*) and the fact that all the elements $a_{ij}^{(k)}$ for $k\ge 1$ are bounded above by 1, we can conclude from Lemma A.2 of Appendix that $\lim_{k\to\infty}s_k=\xi>0$ exists, where $s_k=\prod_{i=1}^{n}d_i^{(k)}e_i^{(k)}$, and consequently that

$$\lim_{k\to\infty}\prod_{i=1}^{n}\sqrt{r_i^{(k)}c_i^{(k)}}=\lim_{k\to\infty}\frac{s_k}{s_{k+1}}=1\,.$$

We conclude that

$$\lim_{k\to\infty}\prod_{i=1}^{n}r_i^{(k)}c_i^{(k)}=1\ \text{ and }\ \lim_{k\to\infty}\sum_{i=1}^{n}\frac{1}{2n}\left(r_i^{(k)}+c_i^{(k)}\right)=1\,. \quad (3.6)$$

Now, since all the elements in $\mathbf{A}^{(k)}$ are less than 1 after the first iteration, each of the two sequences $(r_i^{(k)})_{k\ge1}$ and $(c_j^{(k)})_{k\ge1}$, for all $1\le i,j\le n$, are bounded. Let us introduce the sequence $(\mathbf{v}^{(k)})_{k\ge1}$ of the $2n$-vectors

$$\mathbf{v}^{(k)}=(r_1^{(k)},\dots,r_n^{(k)},c_1^{(k)},\dots,c_n^{(k)})\,,$$

which is also bounded in $\mathbb{R}^{2n}$ of finite dimension. Consider then any convergent subsequence $(\mathbf{v}^{(q)})_q$ and let

$$x_i=\lim_{q\to\infty}r_i^{(q)},\ 1\le i\le n\,,\text{and }y_j=\lim_{q\to\infty}c_j^{(q)},\ 1\le j\le n\,.$$

From (3.6), we can write

$$\prod_{i=1}^{n} x_i y_i = \left\{ \sum_{i=1}^{n} \frac{1}{2n} (x_i + y_i) \right\}^{2n} = 1 ,$$

and since equality holds in the arithmetic-geometric mean inequality only if all the elements are equal, $x_1 = \cdots = x_n = 1 = y_1 = \cdots = y_n$. Therefore any convergent subsequence of the bounded sequence $(\mathbf{v}^{(k)})_{k\geq 1}$ in finite dimensional space must have the same limit (a vector of ones), which implies that the sequence $(\mathbf{v}^{(k)})_{k\geq 1}$ is necessarily convergent and that (3.4) holds.

Since our algorithm fulfills properties (P1*) and (P2), it converges. Additionally, since the computation of the scaling factors in Algorithm 1 is only based on the row and column sums in matrix $\mathbf{A}^{(k)}$ at each iteration, we see that the algorithm is permutation insensitive, and it does not mix information from independent subsets of entries residing in different blocks of a decomposable matrix $\mathbf{A}$. This can be used to show that the scaling matrices also exist (see Corollary A.4 and accompanying discussion in the Appendix) when $\mathbf{A}$ has total support. □

Tracking the computations performed by Algorithm 1 it is straightforward to establish that it preserves symmetry.

COROLLARY 3.2.

1. *If* $\mathbf{A}$ *is symmetric and has support, then Algorithm 1 in the 1-norm builds a sequence of symmetric scalings of* $\mathbf{A}$ *converging to a symmetric doubly stochastic limit.*
2. *If* $\mathbf{A}$ *is symmetric and has total support, then* $\mathbf{A}$ *is symmetrically equivalent to a symmetric doubly stochastic matrix* $\mathbf{S}$*, and Algorithm 1 in the 1-norm builds a convergent sequence of diagonal matrices* $\mathbf{D}^{(k)}$ *such that* $\mathbf{S} = \lim_{k\to\infty} \mathbf{D}^{(k)}\mathbf{A}\mathbf{D}^{(k)}$.

**3.3. Rate of convergence for symmetric matrices.** Let $\mathbf{e}$ be a vector of ones (dimension should be clear) and $\mathcal{D}(\mathbf{x}) = \mathrm{diag}(\mathbf{x})$ for $\mathbf{x} \in \mathbb{R}^n$. Note that $\mathbf{x} = \mathcal{D}(\mathbf{x})\mathbf{e}$ and $\mathcal{D}(\mathcal{D}(\mathbf{x})\mathbf{y}) = \mathcal{D}(\mathbf{x})\mathcal{D}(\mathbf{y}) = \mathcal{D}(\mathbf{y})\mathcal{D}(\mathbf{x})$. When we take the square root of a vector this should be interpreted componentwise (see, e.g., (3.8)). We will also assume that $\mathbf{A} \geq 0$ and that it has no zero rows.

We first treat the case when $\mathbf{A}$ is symmetric. Then the 1-norm version of Algorithm 1 loops the following steps (where $\mathbf{A}^{(0)} = \mathbf{A}$ and $\mathbf{E}^{(0)} = \mathbf{I}$)

$$\mathbf{R} = \mathcal{D}(\mathbf{A}^{(k)}e)^{1/2}, \quad \mathbf{E}^{(k+1)} = \mathbf{R}^{-1}\mathbf{E}^{(k)}, \quad \mathbf{A}^{(k+1)} = \mathbf{R}^{-1}\mathbf{A}^{(k)}\mathbf{R}^{-1}. \tag{3.7}$$

Note that $\mathbf{A}^{(k)} = \mathbf{E}^{(k)}\mathbf{A}\mathbf{E}^{(k)}$ and we can come up with a more compact form of the algorithm as follows. Let $\mathbf{x}_k$ be the vector such that $\mathcal{D}(\mathbf{x}_k) = \mathbf{E}^{(k)}$. Then

$$\begin{aligned}
\mathbf{E}^{(k+1)} &= \mathbf{R}^{-1}\mathbf{E}^{(k)} = \mathcal{D}(\mathbf{E}^{(k)}\mathbf{A}\mathbf{E}^{(k)}e)^{-1/2}\mathbf{E}^{(k)} \\
&= \mathcal{D}(\mathcal{D}(\mathbf{x}_k)\mathbf{A}\mathbf{x}_k)^{-1/2}\mathcal{D}(\mathbf{x}_k) = \mathcal{D}(\mathbf{x}_k)^{-1/2}\mathcal{D}(\mathbf{A}\mathbf{x}_k)^{-1/2}\mathcal{D}(\mathbf{x}_k) \\
&= \mathcal{D}(\mathbf{x}_k)^{1/2}\mathcal{D}(\mathbf{A}\mathbf{x}_k)^{-1/2}.
\end{aligned}$$

In other words, we can carry out the iteration simply working with the vectors $\mathbf{x}_k$. Namely, let $\mathbf{x}_0 = \mathbf{e}$ (say) and form

$$\mathbf{x}_{k+1} = \sqrt{\frac{\mathbf{x}_k}{\mathbf{A}\mathbf{x}_k}}, \tag{3.8}$$

where the division is performed componentwise. If $\mathbf{A}$ is unsymmetric we can replace it with $\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix}$ and extract the row and column scalings from the top and bottom half of $\mathbf{x}_k$.

The reason for introducing the compact form is to allow us to exploit some standard theory for fixed point iteration. In the 1-norm, the balancing problem can be viewed as an attempt to find the positive solution of the nonlinear equation

$$\mathcal{D}(\mathbf{x})\mathbf{A}\mathbf{x} = \mathbf{e}\,. \tag{3.9}$$

This can be rewritten in many ways. In particular, noting that $\mathcal{D}(\mathbf{x})\mathbf{e} = \mathcal{D}(\mathbf{A}\mathbf{x})^{-1}\mathbf{e}$, we can write

$$\mathbf{x} = \mathcal{D}(\mathbf{x})\mathbf{e} = \sqrt{\mathcal{D}(\mathbf{x})^2}\mathbf{e} = \sqrt{\mathcal{D}(\mathbf{x})\mathcal{D}(\mathbf{A}\mathbf{x})^{-1}}\mathbf{e} = \sqrt{\mathcal{D}(\mathbf{A}\mathbf{x})^{-1}\mathbf{x}}\,.$$

In other words (3.8) is a fixed point iteration for solving (3.9). Note that Sinkhorn–Knopp can also be framed in terms of a fixed point iteration for solving (3.9), although some care needs to be taken to ensure that the correct iterates are extracted [**?**].

The consequence of this discussion is that we can prove a result on the asymptotic convergence rate of the algorithm by bounding the norm of the Jacobian of $f(\mathbf{x}) = \sqrt{\mathcal{D}(\mathbf{A}\mathbf{x})^{-1}\mathbf{x}}$ in the environs of a fixed point. We restrict $\mathbf{x}$ so that it lies in the positive cone $\mathbb{R}^n_+$ to ensure that all our diagonal matrices are invertible, and we only take the square roots of positive numbers. Not only is $f(\mathbf{x})$ continuous but it is twice differentiable. Differentiating $f(\mathbf{x})$ term by term, we find that its Jacobian can be written as

$$J(\mathbf{x}) = -\frac{1}{2}\mathcal{D}(\mathbf{x})^{1/2}\mathcal{D}(\mathbf{A}\mathbf{x})^{-3/2}\mathbf{A} + \frac{1}{2}\mathcal{D}(\mathbf{A}\mathbf{x})^{-1/2}\mathcal{D}(\mathbf{x})^{-1/2}. \tag{3.10}$$

To establish the asymptotic rate of convergence, we want to bound $\|J(\mathbf{x})\|$ at the fixed point $\mathbf{x}_*$. Substituting the identity $\mathcal{D}(\mathbf{A}\mathbf{x}_*) = \mathcal{D}(\mathbf{x}_*)^{-1}$ into (3.10) gives

$$\begin{aligned} J(\mathbf{x}_*) &= -\frac{1}{2}\mathcal{D}(\mathbf{x}_*)^{1/2}\mathcal{D}(\mathbf{x}_*)^{3/2}\mathbf{A} + \frac{1}{2}\mathcal{D}(\mathbf{A}\mathbf{x}_*)^{1/2}\mathcal{D}(\mathbf{x}_*)^{-1/2} \\ &= \frac{1}{2}\left(\mathbf{I} - \mathcal{D}(\mathbf{x}_*)^2\mathbf{A}\right) = \frac{1}{2}\mathcal{D}(\mathbf{x}_*)\left(\mathbf{I} - \mathcal{D}(\mathbf{x}_*)\mathbf{A}\mathcal{D}(\mathbf{x}_*)\right)\mathcal{D}(\mathbf{x}_*)^{-1} \\ &= \frac{1}{2}\mathcal{D}(\mathbf{x}_*)(\mathbf{I} - \mathbf{P})\mathcal{D}(\mathbf{x}_*)^{-1}, \end{aligned}$$

where $\mathbf{P}$ is symmetric and stochastic. In a neighbourhood of $\mathbf{x}_*$ we can bound the asymptotic rate of convergence of the iteration by the largest eigenvalue of $(\mathbf{I}-\mathbf{P})/2$. Since the spectrum of $\mathbf{P}$ lies in $[-1, 1]$, the bound is $(1 - \lambda_{\min}(\mathbf{P}))/2$.

We can couple our observations with the convergence results from Section 3 to make a strong statement about the performance of the algorithm.

THEOREM 3.3. *Let $\mathbf{A} \geq 0$ be symmetric and be fully indecomposable. Then for any vector $\mathbf{x}_0 > 0$, the iteration (3.8) will converge to the unique positive vector $\mathbf{x}_*$ such that $\mathcal{D}(\mathbf{x}_*)\mathbf{A}\mathcal{D}(\mathbf{x}_*) = \mathbf{P}$ is doubly stochastic and the convergence is asymptotically linear at the rate $(1 - \lambda_{\min}(\mathbf{P}))/2 < 1$.*

*Proof.* We already know that (3.8) converges from Theorem 3.1. The uniqueness of $\mathbf{x}_*$ is well known (see, for example, Lemma 4.1 of Knight [**?**]).

Since $\mathbf{A}$ is fully indecomposable, so is $\mathbf{P}$. Such a doubly stochastic matrix is primitive [**?**], and hence it has a single simple eigenvalue of modulus 1. Therefore

$\lambda_{\min}(\mathbf{P}) > -1$ and the asymptotic rate of convergence, established above, is bounded below one. □

Note that Theorem 3.1 establishes the convergence of (3.8) for matrices with total support. If a matrix has total support but is not fully indecomposable then it can be permuted symmetrically into a block diagonal matrix where each block is irreducible [**?**, Lemma 3.1]. Now if an irreducible block is also fully indecomposable then we can apply Theorem 3.3 to it. Suppose that a block is irreducible but not fully indecomposable. It must be permutable to the form (3.1). Since no zero block in an irreducible matrix can be permuted symmetrically to lie in the bottom left hand corner, the lack of indecomposability manifests itself in a diagonal block. That is, there is a symmetric permutation of the block to $\begin{pmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{Z} \end{pmatrix}$ where $\mathbf{B}$ is fully indecomposable. Since $\mathbf{A}$ has total support, $\mathbf{Z} = \mathbf{0}$. For a matrix with this structure, we can still give the rate of convergence: it is covered by a general convergence result for (3.8) on unsymmetric matrices, which we establish next.

**3.4. Rate of convergence for unsymmetric matrices.** Much of our analysis carries through if $\mathbf{A}$ is unsymmetric and we work with $\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix}$ instead: using (3.7) on this matrix is exactly the same as running Algorithm 1 on $\mathbf{A}$. At a fixed point, the Jacobian matrix can be written as

$$J(\mathbf{x}_*) = \frac{1}{2}\mathcal{D}(\mathbf{x}_*)\left(\mathbf{I} - \begin{pmatrix} \mathbf{0} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{pmatrix}\right)\mathcal{D}(\mathbf{x}_*)^{-1}, \tag{3.11}$$

where $\mathbf{P}$ is doubly stochastic but may be unsymmetric. There is a snag, though. Notice that

$$\begin{pmatrix} \mathbf{0} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{e} \\ -\mathbf{e} \end{pmatrix} = \begin{pmatrix} -\mathbf{e} \\ \mathbf{e} \end{pmatrix},$$

from which we conclude that $-1$ is an eigenvalue of $J(\mathbf{x}_*)$, and we can no longer apply the contraction mapping theorem. Nor is the fixed point unique: if $\mathbf{P} = \mathbf{DAE}$, then $\alpha\mathbf{D}$ and $\mathbf{E}/\alpha$ give another solution.

A similar problem arises in trying to apply the contraction mapping theorem in the analysis of the Sinkhorn–Knopp algorithm [**?**]. Here, as there, this difficulty is overcome once we note that we do not really care which fixed point we end up at, just that we get closer and closer to the set of fixed points. But we know this happens: we have a global convergence result (Theorem 3.1).

THEOREM 3.4. *Let*

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{0} \end{pmatrix}$$

*where* $\mathbf{B} \in \mathbb{R}^{n\times n}$ *is nonnegative and fully indecomposable, and let* $\mathbf{Q}$ *be the doubly stochastic matrix one gets from diagonally balancing* $\mathbf{B}$.

*The iteration* (3.8) *converges linearly for all initial vectors* $\mathbf{x}_0 > 0$ *with asymptotic rate of convergence* $\rho = (1+\sigma_2(\mathbf{Q}))/2$ *where* $\sigma_2(\mathbf{Q})$ *is the second largest singular value of* $\mathbf{Q}$.

As with Theorem 3.3, we can extend Theorem 3.4 to cover unsymmetric matrices which are not fully indecomposable but have total support by working with a block diagonal permutation.

*Proof.* [Theorem 3.4] Convergence of the algorithm for any starting vector is guaranteed by Theorem 3.1. Let us assume that the limit point is $\mathbf{x}_* = [\mathbf{r}^T \ \mathbf{c}^T]^T$. Note that the set of fixed points of (3.8) can be written as

$$S = \left\{ \begin{pmatrix} \beta\mathbf{r} \\ \frac{1}{\beta}\mathbf{c} \end{pmatrix}, \ \beta \in \mathbb{R}_+ \right\}.$$

Let $\epsilon > 0$ and choose $K$ so that for $k > K$, $\|\mathbf{x}_k - \mathbf{x}\|_* < \epsilon$ where we choose (for reasons we will explain)

$$\|\mathbf{x}\|_* = \sqrt{\frac{\mathbf{x}^T \mathcal{D}(\mathbf{x}_*)^{-2}\mathbf{x}}{2n}} \, .$$

We will show that

$$\min_{\mathbf{s}\in S} \|\mathbf{x}_{k+1} - \mathbf{s}\|_* \leq \rho \min_{\mathbf{s}\in S} \|\mathbf{x}_k - \mathbf{s}\|_* + \tau_k \, , \tag{3.12}$$

where $\tau_k = \mathcal{O}(\epsilon^2)$, and hence we can infer the desired convergence rate for small enough values of $\epsilon$. Note that $\mathbf{x}_*$ may not be the nearest element to $\mathbf{x}_k$ in $S$: the nearest point is

$$\mathbf{s}_k = \begin{pmatrix} \alpha\mathbf{r} \\ \frac{1}{\alpha}\mathbf{c} \end{pmatrix}$$

where $\alpha = 1 + \epsilon_\alpha$ and $\epsilon_\alpha = \mathcal{O}(\epsilon)$. Now

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k) = f(\mathbf{x}_* + \mathbf{p}) = \mathbf{x}_* + J(\mathbf{x}_*)\mathbf{p} + \mathbf{q} \, ,$$

where $\|\mathbf{p}\|_* < \epsilon$ and $\|\mathbf{q}\|_* = \mathcal{O}(\epsilon^2)$. From (3.11) we can deduce that the eigenvalues of $J(\mathbf{x}_*)$ are

$$\lambda_i = \frac{1}{2}(1 + \sigma_i(\mathbf{Q})), \quad \lambda_{i+n} = \frac{1}{2}(1 - \sigma_{n+1-i}(\mathbf{Q})), \quad i = 1, 2, \ldots, n,$$

where $\sigma_i(\mathbf{Q})$ is the $i$th singular value of $\mathbf{Q}$. Since $\mathbf{B}$ is fully indecomposable, then so is $\mathbf{Q}$, and it is primitive. Hence $1 = \sigma_1(\mathbf{Q}) > \sigma_2(\mathbf{Q}) \geq \cdots \geq \sigma_n(\mathbf{Q}) \geq 0$. Let $\mathbf{v}_i$ be the eigenvector of $J(\mathbf{x}_*)$ associated with $\lambda_i$ such that $\|\mathbf{v}_i\|_* = 1$ and write

$$\mathbf{p} = \sum_{i=1}^{2n} \mu_i \mathbf{v}_i \, .$$

From (3.11) we know that $J(\mathbf{x}_*)$ is similar to a symmetric matrix. Our choice of norm is motivated by our need to control the size of $J(\mathbf{x}_*)\mathbf{p}$ for all $k > K$. Since the $\mathbf{v}_i$ form an orthonormal set with respect to the inner product that induces our norm, $|\mu_i| \leq \|\mathbf{p}\|_* < \epsilon$, for all $i$.

Noting that $\lambda_1 = 1$, $\mathbf{v}_1 = [\mathbf{r}^T \ -\mathbf{c}^T]^T$, $\lambda_{2n} = 0$ and $\mathbf{v}_{2n} = \mathbf{x}_*$, we have

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_* + \mu_1\mathbf{v}_1 + \sum_{i=2}^{2n-1} \lambda_i\mu_i\mathbf{v}_i + \mathbf{q} \\ &= \begin{pmatrix} (1+\mu_1)\mathbf{r} \\ (1-\mu_1)\mathbf{c} \end{pmatrix} + \sum_{i=2}^{2n-1} \lambda_i\mu_i\mathbf{v}_i + \mathbf{q} \\ &= \begin{pmatrix} \gamma\mathbf{r} \\ \mathbf{c}/\gamma \end{pmatrix} + \begin{pmatrix} 0 \\ \eta\mathbf{c} \end{pmatrix} + \sum_{i=2}^{2n-1} \lambda_i\mu_i\mathbf{v}_i + \mathbf{q} \end{aligned}$$

where $\gamma = (1+\mu_1) = 1 + \mathcal{O}(\epsilon)$ and $\eta = -\mu_1^2/(1+\mu_1)$.

In order to establish (3.12), we need to relate the size of the sum in our expression for $\mathbf{x}_{k+1}$ to $\min_{\mathbf{s}\in S} \|\mathbf{x}_k - \mathbf{s}\|_* = \|\mathbf{x}_k - \mathbf{s}_k\|_*$. Since

$$\mathbf{x}_k - \mathbf{s}_k = \mathbf{p} + \mathbf{x}_* - \mathbf{s}_k = \mathbf{p} + \begin{pmatrix} (1-\alpha)\mathbf{r} \\ (1-1/\alpha)\mathbf{c} \end{pmatrix} = \sum_{i=1}^{2n} \mu_i \mathbf{v}_i + \frac{1-\alpha^2}{2\alpha}\mathbf{v}_1 - \frac{(\alpha-1)^2}{2\alpha}\mathbf{v}_{2n},$$

we have

$$\left\| \sum_{i=2}^{2n-1} \lambda_i \mu_i \mathbf{v}_i \right\|_* \le \lambda_2 \sqrt{\sum_{i=2}^{2n-1} \|\mu_i \mathbf{v}_i\|_*^2} = \lambda_2 \left\| \sum_{i=2}^{2n-1} \mu_i \mathbf{v}_i \right\|_* \le \lambda_2 \|\mathbf{x}_k - \mathbf{s}_k\|_* ,$$

and letting

$$\tau_k = \left\| \begin{pmatrix} 0 \\ \eta\mathbf{c} \end{pmatrix} + \mathbf{q} \right\|_* = \mathcal{O}(\epsilon^2) ,$$

we are done. □

Our rate of convergence result allows a direct comparison with other scaling algorithms. In particular, it is shown [**?**] that the Sinkhorn–Knopp algorithm also converges linearly with an asymptotic rate of convergence of $\sigma_2(\mathbf{Q})$. Note that since $\sigma_2(\mathbf{Q}) < 1$, we have $\rho > \sigma_2(\mathbf{Q})$. Hence the bound on the asymptotic rate of convergence (and empirical evidence suggests that it is generally a sharp bound) is **necessarily** larger than the corresponding bound for the Sinkhorn–Knopp algorithm.

However, for symmetric matrices the comparison is not so clear cut. The asymptotic rates are $\lambda_2(\mathbf{P})$ for the Sinkhorn–Knopp algorithm versus $(1 - \lambda_{\min}(\mathbf{P}))/2$ for Algorithm 1. For sparse matrices we can expect both of these factors to be close to one (especially if the matrix is close to being reducible). But note that when $|\mathbf{A}|$ is symmetric positive definite, the rate of convergence of Algorithm 1 is guaranteed to be less than $1/2$, since $\mathbf{P}$ is symmetric positive definite, too.

Often, though, we only need a modest degree of scaling to induce beneficial effects in linear solvers. Here, the fact that Algorithm 1 retains symmetry is a distinct advantage: even though a symmetric matrix balanced by the Sinkhorn–Knopp algorithm will (in the limit) be symmetric, the intermediate scalings lose this property.

A number of balancing algorithms [**?**, **?**] have been developed based on Newton iterations for solving (3.9). While these retain symmetry in the intermediate scalings and can achieve quadratic convergence asymptotically, they can behave erratically when the iterates are far from being truly balanced, a phenomenon we have not seen with Algorithm 1. Typically, we have found that this erratic behaviour occurs[1] when the linear systems in the Newton step are singular (but consistent) and may be due to unwanted elements of the null-space being included in the early iterates. In future work, it may be worth investigating whether Newton-based methods can be employed in the applications we discuss in our numerical experiments. One limitation of these methods is that they are not designed for $\infty$-norm scaling.

**4. Numerical experiments.** We have implemented the proposed algorithm in Matlab (version R2009a), and compared it against our implementation of Bunch's and Sinkhorn–Knopp algorithms. We have experimented with a set of matrices obtained

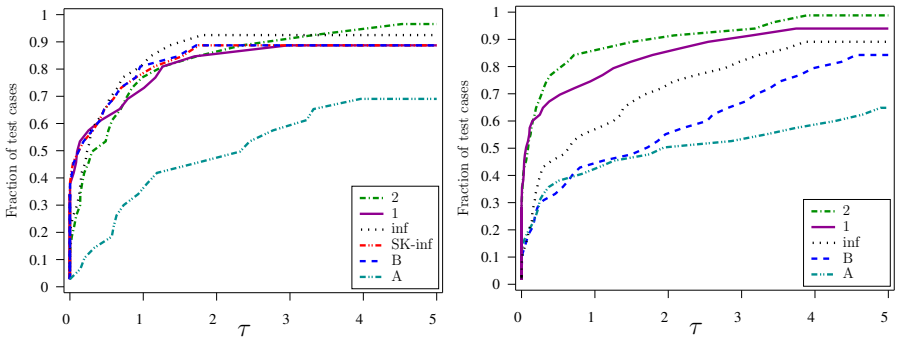
[1] Seen for some matrices from the GHS_indef group from the UFL collection.

from the University of Florida Sparse Matrix Collection (UFL) available at http://www.cise.ufl.edu/research/sparse/matrices/. The matrices in our data set satisfy the following properties: real, $1000 \leq n \leq 121000$, $2n \leq \text{nnz} \leq 1790000$, without explicit zeros, fully indecomposable, not a matrix with nonzeros only in the set $\{-1, 1\}$. There were a total of 224 such matrices at the time of experimentation. In seven out of those 224 matrices, Matlab's `condest` function failed to compute an estimated condition number in a reasonable amount of time, on three matrices `condest` gave `inf` as the estimated condition number, and one matrix was reported to be rank-deficient at the UFL collection. Therefore, we give results on 213 matrices, among which 64 are unsymmetric, 46 are symmetric positive definite (SPD), and 103 are symmetric but not positive definite.

We present two different sets of experiments. In the first one, we compare different scaling algorithms. In the second one, we investigate the merits of using the proposed scaling algorithm in solving linear systems with a direct method. In terms of convergence, there is little difference in theory which $p$-norm we choose to scale by in Algorithm 1 (apart from the $\infty$-norm). But to investigate the potential of the algorithm in practice we have tested the both the 1-norm and 2-norm versions alongside the $\infty$-norm. For comparison, we have also tested the 1- and 2-norm versions of the Sinkhorn–Knopp algorithm as well as Bunch's $\infty$-norm algorithm. For an unsymmetric matrix $\mathbf{A}$, we apply Bunch's algorithm to $\begin{pmatrix} \mathbf{0} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{pmatrix}$ which is equivalent to applying the Sinkhorn-Knopp algorithm using the $\infty$-norm (SK-inf). For symmetric matrices (both general and SPD), we did not include results with SK-inf, as this alternative loses the symmetry. For SPD matrices, we also use the reciprocal of the square root of diagonal entries as the scaling matrices (on both sides). We refer to this algorithm as `rdiag`.

**4.1. Condition numbers and iterations.** We use the performance profiles [**?**] to assess the effect of the scaling algorithms on the condition number. In order to generate these profiles, we compute the condition numbers of a given matrix and its scaled versions (with different scaling algorithms). This gives us a set of condition estimates $\{\chi_1, \ldots, \chi_p\}$ corresponding to $p$ different scaling algorithms (the condition number of the original matrix can be thought of resulting from a scaling algorithm) for a given matrix. We then compute $\{\chi_1/\chi, \ldots, \chi_p/\chi\}$ where $\chi$ is the smallest value in the set. We take the logarithms of these ratios to avoid large numbers, and designate the resulting number as the performance of an algorithm on the given matrix. Then, we measure the percentage of the number of matrices in which the performance of an algorithm is less than or equal to a given number $\tau$. Plotting these measurements gives the performance profiles shown in Fig. 4.1 (the three subfigures correspond to the three classes of matrices). In other words, for a given value of $\tau$, a plot in Fig. 4.1 shows the fraction of the number of matrices for which the corresponding scaling algorithm obtains a condition estimate within $e^\tau$ of the smallest (among all the condition estimates). A high fraction indicates that a given algorithm regularly keeps the estimated condition number under control and hence indicates that algorithm's effectiveness (particularly if the fraction is high for small values of $\tau$).

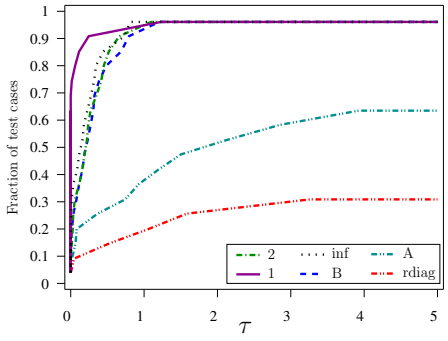
In these experiments, all the iterative scaling methods are run with an error tolerance of 1.0e-4 and with the number of iterations limited to the matrix dimension. All condition numbers are computed using Matlab's `condest` function. For clarity, the profiles for Sinkhorn–Knopp algorithm in 1- and 2-norms are not shown. Generally they agreed with the proposed algorithm in the corresponding norms to at least four




(a) Unsymmetric matrices




(b) General symmetric matrices




(c) SPD matrices

FIG. 4.1. *Performance profiles for the condition number estimates of unsymmetric matrices 4.1(a), general symmetric matrices 4.1(b), and SPD matrices 4.1(c). Horizontal axis in each subfigure corresponds to* $\tau$. `A` *marks the condition number estimate of the original matrix;* `B` *marks that of Bunch's algorithm;* `inf`, `1`, *and* `2` *mark that of the proposed algorithm with* $\infty$*-, 1-, and 2-norms,* `SK-inf` *marks that of Sinkhorn–Knopp with* $\infty$*-norm and* `rdiag` *marks that of the reciprocal square root of the diagonal scaling for SPD matrices.*

significant digits—exceptions to this rule tend to coincide with cases where either the proposed algorithm or Sinkhorn–Knopp had not converged in $n$ iterations.

In all of the subfigures of Fig. 4.1, the profiles of the scaling algorithms (with the single exception of `rdiag`) are higher than the profile representing no scaling (shown with `A`), for any range of $\tau$. Therefore, one concludes that the scaling algorithms improve the estimated condition number in an overwhelming majority of cases. For unsymmetric matrices, the algorithms that work with the $\infty$-norm (`inf`, `SK-inf`, and `B`) outperform (or, have higher profiles than) the 1- and 2-norm scalings until $\tau > 3.2$ after which the 2-norm scaling takes over. For general symmetric matrices, a clear pattern is seen in Fig. 4.1(b). This time, the 2-norm scaling has the highest performance profile, marginally beating the 1-norm scaling algorithm which in turn appears superior to the proposed algorithm in the $\infty$-norm over all shown ranges of $\tau$. Bunch's algorithm is ranked most poorly. For the SPD matrices, all algorithms (except `rdiag`) are almost equal if one considers estimated condition numbers that are within $e$ of the best, as the fraction of the test cases that are below $\tau = e$ are above 0.90.

We give further details in Table 4.1 which shows that all of the scaling algorithms

TABLE 4.1

*The numbers of matrices in which the estimated condition number of the scaled matrix is larger than (the first half of the table) and smaller than (the second half) the condition number (estimated) of the original matrix by differing amounts, for the unsymmetric, symmetric, and SPD matrices in the data set.*

| | Unsymmetric (64) | | | | | Symmetric (103) | | | | SPD (46) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ratio | SK-inf | B | A-inf | A-1 | A-2 | B | A-inf | A-1 | A-2 | $\sqrt{d}$ | B | A-inf | A-1 | A-2 |
| ≥1.0e+3 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 21 | 0 | 0 | 0 | 0 |
| ≥1.0e+2 | 0 | 0 | 0 | 3 | 1 | 1 | 0 | 1 | 0 | 25 | 0 | 0 | 0 | 0 |
| ≥1.0e+1 | 1 | 1 | 0 | 6 | 4 | 12 | 1 | 2 | 2 | 32 | 1 | 0 | 0 | 0 |
| >1.0e+0 | 10 | 9 | 6 | 12 | 10 | 30 | 24 | 18 | 19 | 40 | 9 | 6 | 8 | 10 |
| ≤1.0e-3 | 11 | 11 | 10 | 11 | 10 | 18 | 20 | 25 | 27 | 0 | 10 | 10 | 11 | 10 |
| ≤1.0e-4 | 6 | 6 | 8 | 8 | 8 | 11 | 13 | 19 | 23 | 0 | 7 | 7 | 7 | 7 |
| ≤1.0e-5 | 5 | 5 | 6 | 5 | 4 | 8 | 8 | 11 | 14 | 0 | 3 | 3 | 3 | 3 |
| ≤1.0e-6 | 5 | 5 | 4 | 3 | 4 | 5 | 5 | 8 | 8 | 0 | 3 | 3 | 3 | 3 |

TABLE 4.2

*The statistics on the numbers of iterations of Sinkhorn–Knopp algorithm in 1- and 2-norms (SK-1 and SK-2) and the proposed algorithm in 1-, 2-, and ∞-norms (A-1, A-2, and A-∞) to reach an error tolerance $\varepsilon = 1.0\text{e-}4$ with a limit of $n$ in the number of iterations.*

| stat. | $n$ | SK-1 | | SK-2 | | A-1 | | A-2 | | A-∞ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | its | its/$n$ | its | its/$n$ | its | its/$n$ | its | its/$n$ | its | its/$n$ |
| unsymmetric (64) | | | | | | | | | | | |
| min | 1000 | 1 | 0.000 | 47 | 0.015 | 1 | 0.000 | 6 | 0.000 | 2 | 0.000 |
| max | 115967 | 84617 | 1.000 | 84617 | 1.000 | 84617 | 1.000 | 84617 | 1.000 | 19 | 0.000 |
| geomean | 6888 | 1057 | 0.153 | 1615 | 0.234 | 776 | 0.113 | 1045 | 0.152 | 6 | 0.001 |
| general symmetric (103) | | | | | | | | | | | |
| min | 1224 | 8 | 0.000 | 1 | 0.000 | 3 | 0.000 | 1 | 0.000 | 2 | 0.000 |
| max | 93279 | 11870 | 0.916 | 14443 | 1.000 | 10307 | 0.328 | 15343 | 1.000 | 19 | 0.011 |
| geomean | 12096 | 310 | 0.026 | 646 | 0.053 | 52 | 0.004 | 67 | 0.006 | 7 | 0.001 |
| SPD (46) | | | | | | | | | | | |
| min | 1083 | 73 | 0.001 | 46 | 0.002 | 7 | 0.000 | 3 | 0.000 | 2 | 0.000 |
| max | 102158 | 8205 | 1.000 | 10795 | 1.000 | 17 | 0.013 | 18 | 0.012 | 17 | 0.008 |
| geomean | 10349 | 471 | 0.046 | 920 | 0.089 | 13 | 0.001 | 10 | 0.001 | 4 | 0.000 |

(except `rdiag`) improve the condition number in most of the instances (below the row corresponding to 1.0e+0). In fact, `rdiag` increases the condition number of most matrices in our data.

As seen from these numbers, using the proposed algorithm in 1- and 2-norms for unsymmetric matrices can cause ill-conditioning (problem ids at the UFL collection are 287, 289, 816, 1200, and 1433). In matrices which are close to being decomposable, scaling algorithms "want" to zero out certain elements. This cannot be done exactly and the resultant lop-sided scalings cause the condition number to go through the roof (up by a factor of $10^{34}$ in extremis). We do not advocate scaling as a panacea but, particularly in the symmetric case (when, if there is a full diagonal, we can expect Sinkhorn–Knopp type algorithms to rapidly converge) the scaling is valuable. And note that the negative phenomenon of wildly worse conditioning is not present in ∞-norm scalings for unsymmetric matrices as the pressure to remove entries disappears.

We present some statistical indicators (the minimum, the maximum, and the geometric mean) of the number of iterations with the same setting as before for the proposed algorithm in 1-, 2-, and ∞-norms and Sinkhorn–Knopp algorithm in 1- and 2-norms in Table 4.2. In this table, $n$ is the size of the matrices, "its" is the number of iterations, and "its/$n$" is the ratio of these two quantities. For the latter column, we computed the ratio for each matrix and then take the minimum, the maximum, and

the geometric mean. As seen from the numbers, the proposed algorithm in $\infty$-norm converges rapidly, displaying a geometric mean of iterations below 10. The 1-norm and 2-norm scaling algorithms have large number of iterations (see geometric means) for unsymmetric matrices. Those numbers for the Sinkhorn–Knopp algorithm are even larger. The 2-norm variant of both algorithms have larger geometric means, and hence slower convergence than their 1-norm variants. For the symmetric and symmetric positive definite matrices, the number of iterations of the proposed algorithm in the 1- and 2-norms are better than the Sinkhorn–Knopp variants.

**4.2. Uses of the methods in a direct solver.** We investigate the use of the proposed scaling algorithm in the solution of linear systems with direct methods. We do not perform experiments with SPD matrices. We use the solver MUMPS [**?**, **?**] version 4.10 in our tests with its default parameter settings except for the following. For the unsymmetric matrices, if there are zeros in the diagonal, we find a maximum traversal using `dmperm` of Matlab and permute the matrix to put that transversal into the diagonal before calling MUMPS and set the related parameter ICNTL(6) to 0 in MUMPS. For the general symmetric matrices, we set the parameter ICNTL(12) to 2 to use the compressed ordering strategies [**?**] along with a columnwise permutation option ICNTL(6)=1, equivalent in essence to `dmperm` of Matlab. Before the numerical factorization, MUMPS allocates some memory with amount equal to the estimated one plus $X\%$ more where $X$ is specified by the memory relaxation parameter ICNTL(14). MUMPS returns an error message if the actual memory requirements that arise during numerical factorization pass the allocated one. Extra allocation of memory is necessary when pivoting is used to ensure numerical stability. A scaling strategy is deemed to be better if it increases the number of successful runs (by reducing pivoting and hence its effects on memory and work). No other warnings or error messages were emitted by MUMPS for the matrices in the data set.

As the number of iterations of the proposed algorithm can be high (see Table 4.2), we recommend its use with a fixed and moderate number of iterations rather than to full convergence. In order to suggest a solid scaling approach, we have investigated different strategies specified below by three integers $[i_1, i_2, i_3]$. Each strategy proceeds in three phases: $i_1$ steps of $\infty$-norm scaling; then $i_2$ steps of 1- or 2-norm scaling; and finally $i_3$ steps of $\infty$-norm scaling. It is understood that each phase continues using the scaling factors found so far, and if convergence occurs within a phase then the next phase is started. We use the $\infty$-norm scaling in the first and third phase as a smoother to reduce each nonzero entry below one. The 1- and 2-norm scaling iterations are two lead to different scaled matrices therefore we do not use one after another. The results are presented in Table 4.3. In this table, the scaling strategies are represented as $[i_1, i_2^{(1)}, i_3]$ or $[i_1, i_2^{(2)}, i_3]$ where $i_2^{(1)}$ and $i_2^{(2)}$ designate the number of 1-norm or 2-norm iterations performed in phase 2 of the scaling strategy. We also include results with Bunch's algorithm, Sinkhorn–Knopp algorithm with $\infty$-norm (very close to that of Bunch's), and Sinkhorn–Knopp algorithm with 10 iterations of 1-norm scaling. We give results with different levels of memory relaxation parameter. We note that MUMPS can run to completion for all matrices in our data set with with one of the scaling strategies with ICNTL(14)= 40. Therefore any error message returned by MUMPS signifies numerical difficulties.

We comment more on symmetric matrices, as the proposed algorithm is particularly motivated for this case. As seen in the right half of Table 4.3, the proposed algorithm requires only a few iterations to reduce the number of unsuccessful runs from 18 (with ICNTL(14)=10) and 9 (with ICNTL(14)=80). In general, increasing

TABLE 4.3

*The effects of different scaling strategies on the direct solver MUMPS version 4.10. A strategy is said to be unsuccessful (usuc.), if MUMPS returns an error message with the settings described in the text, where usuc(X) gives the number of erroneous cases with ICNTL(14)=X. The geometric mean of the ratio of the condition number estimate of the scaled matrix to the original one is given in the column* $\frac{cnd(DAE)}{cnd(A)}$.

| | unsymmetric matrices | | | general symmetric matrices | | |
|---|---|---|---|---|---|---|
| strategy | usuc.(10) | usuc.(80) | $\frac{cnd(DAE)}{cnd(A)}$ | usuc.(10) | usuc.(80) | $\frac{cnd(DAE)}{cnd(A)}$ |
| no-scaling | 4 | 2 | 9.99e-01 | 18 | 9 | 1.00e+00 |
| $[1,0,0]$ | 3 | 1 | 6.15e-02 | 11 | 6 | 1.25e-01 |
| $[3,0,0]$ | 3 | 1 | 4.53e-02 | 9 | 3 | 4.31e-02 |
| $[10,0,0]$ | 3 | 1 | 4.56e-02 | 9 | 2 | 4.00e-02 |
| $[0,3^{(1)},0]$ | 2 | 0 | 3.05e-02 | 4 | 0 | 1.48e-02 |
| $[1,3^{(1)},0]$ | 2 | 0 | 3.13e-02 | 4 | 0 | 1.39e-02 |
| $[1,3^{(1)},1]$ | 2 | 0 | 3.25e-02 | 4 | 0 | 1.68e-02 |
| $[1,3^{(1)},3]$ | 2 | 0 | 3.63e-02 | 5 | 0 | 2.08e-02 |
| $[0,3^{(2)},0]$ | 3 | 0 | 3.87e-02 | 7 | 3 | 2.18e-02 |
| $[1,3^{(2)},0]$ | 2 | 0 | 3.57e-02 | 7 | 3 | 1.97e-02 |
| $[1,3^{(2)},1]$ | 2 | 0 | 3.90e-02 | 7 | 2 | 2.27e-02 |
| $[1,3^{(2)},3]$ | 2 | 0 | 3.93e-02 | 9 | 2 | 2.61e-02 |
| $[0,10^{(1)},0]$ | 2 | 0 | 3.12e-02 | 3 | 0 | 1.71e-02 |
| $[1,10^{(1)},0]$ | 2 | 0 | 3.11e-02 | 3 | 0 | 1.69e-02 |
| $[1,10^{(1)},1]$ | 2 | 0 | 3.33e-02 | 3 | 0 | 1.88e-02 |
| $[0,10^{(2)},0]$ | 3 | 0 | 3.54e-02 | 5 | 1 | 1.57e-02 |
| $[1,10^{(2)},0]$ | 3 | 0 | 3.31e-02 | 4 | 1 | 1.63e-02 |
| $[1,10^{(2)},1]$ | 3 | 0 | 3.43e-02 | 4 | 1 | 1.74e-02 |
| $[1,100^{(1)},0]$ | 2 | 0 | 1.40e-03 | 3 | 0 | 1.08e-02 |
| $[1,100^{(2)},0]$ | 2 | 0 | 3.46e-02 | 3 | 0 | 1.62e-02 |
| Bunch | 5 | 1 | 4.13e-02 | 10 | 1 | 7.54e-02 |
| SKInf | 5 | 1 | 4.66e-02 | | | |
| SK10 | 2 | 1 | 3.39e-02 | | | |

the number of iterations improves success rate. The strategies with the 1-norm iterations are more effective than those with the 2-norm iterations (compare for example the strategies with $[\cdot,3^{(1)},\cdot]$ and $[\cdot,3^{(2)},\cdot]$). Most of the time, the condition number estimates are better with the scaling strategies having 1-norm iterations (we note that the Matlab's estimator works in the 1-norm, so this may also have some effect). Bunch's algorithm is also very effective for symmetric matrices in reducing the unsuccessful runs with ICNTL(14)=80; but not that successful with ICNTL(14)=10. For the unsymmetric matrices, the strategies with the 1-norm iterations are better than those with the 2-norm iterations in reducing the condition estimates and also marginally better in reducing the unsuccessful runs. Bunch's and SK$\infty$ algorithm fail for almost the same number of matrices as with no-scaling (even one more with ICNTL(14)=10). We conclude that the scaling algorithms (proposed and existing ones) are useful in reducing the need for pivoting so that memory requirements (and work) during numerical factorization are kept close to a priori estimates. In particular, strategies $[0,3^{(1)},0]$ or $[1,3^{(1)},0]$ are viable for symmetric and unsymmetric matrices, both in sequential and parallel computing environments. Equivalent SK iterations are also viable for unsymmetric matrices. For symmetric matrices, Bunch's algorithm has merits in sequential computing environments with relatively loose memory limits for the direct solver, but would be harder to parallelize.

**5. Conclusion.** We presented an iterative algorithm which scales the $\infty$-norm of the rows and columns of a matrix to 1. The important features of the proposed algorithm are the following: it preserves symmetry; it is permutation independent; and it has fast linear convergence with an asymptotic rate of 1/2. We discussed the extension to the 1-norm in detail. Again, the algorithm preserves symmetry and is permutation independent. From the various theorems collecting the convergence analysis results in the different norms, it can be seen that the assumptions are much more restrictive and complicated in their combinatorial aspects in the 1-norm as opposed to the $\infty$-norm. In the 1-norm, we have established that convergence depends on the nonzero structure of the matrix in much the same way as other Diagonal Product Increasing algorithms, such as Sinkhorn–Knopp; but that for symmetric matrices our algorithm can accelerate this convergence. This is confirmed in experimental results.

The rates of convergence in the particular case of our algorithm, fixed and fast in the $\infty$-norm, problem-dependent and potentially much slower in the 1-norm, also illustrate the strong difference between the two. But whatever norm we choose, we have numerical evidence for the algorithm's potential to reduce the condition number of a matrix. Our numerical experiments demonstrate that the proposed algorithm can be helpful for a direct solver in the sense that it reduces the need for numerical pivoting. In particular, we have shown that one step of $\infty$-norm scaling followed by a few steps of 1- or 2-norm scaling is often good enough. We have also experimentally demonstrated that the proposed algorithm usually converges faster than Sinkhorn–Knopp scaling method for symmetric matrices.

**Acknowledgements and availability.** We thank Patrick R. Amestoy, Iain S. Duff, Nick Gould, and Mario Arioli for commenting on an earlier draft. We also wish to thank the referees for their detailed reports that helped us improve the manuscript, and Stanley Eisenstat for his very constructive editorial and technical comments.

The Matlab implementations of all scaling algorithms discussed in this paper and an MPI-based parallel implementation of the proposed scaling algorithm are available at `http://perso.ens-lyon.fr/~bucar/codes.html`. The MPI-based parallel version is incorporated in MUMPS [**?**, **?**] (since version 4.8) for scaling distributed input matrices. The sequential Fortran implementation of the proposed scaling algorithm is available as `MC77` in the HSL Mathematical Software Library.

**Appendix A. General convergence results in the 1-norm.** We extend partly the general convergence theorem of Parlett and Landis [**?**] regarding the convergence of "Diagonal Product Increasing" (DPI) algorithms. We show that some of their hypothesis can be weakened while keeping the same results, and that under some stronger hypothesis, their general convergence result can be strengthened too. Our demonstrations here follow very closely the discussions by Parlett and Landis [**?**, Theorem 1, pages 63–68], and improves only partially the already very general scope of their theoretical analysis. We have just included for completeness all the material about our extensions, since they cover directly the specific case of our Algorithm 1 in the 1-norm. We hope that the demonstrations here might be useful to prove the convergence of scaling alternatives.

For the notations, consider an iterative scaling algorithm producing the iterates (3.2) from Section 3.2, that is $\mathbf{A}^{(k)} = \left(a_{ij}^{(k)}\right) = \mathbf{D}^{(k)}\mathbf{A}\mathbf{E}^{(k)}$, where $\mathbf{D}^{(k)} = \operatorname{diag}\left(d_1^{(k)}, \ldots, d_n^{(k)}\right)$ and $\mathbf{E}^{(k)} = \operatorname{diag}\left(e_1^{(k)}, \ldots, e_n^{(k)}\right)$, and where $\mathbf{D}^{(0)} = \mathbf{E}^{(0)} = \mathbf{I}$. Let $r_i^{(k)}$ for $i = 1, \ldots, n$, and $c_j^{(k)}$ for $j = 1, \ldots, n$, denote the 1-norm of rows and columns

respectively. Assuming $\mathbf{A} \geq 0$, the row and column sums are given as $r_i^{(k)} = \sum_{j=1}^n a_{ij}^{(k)}$ and $c_j^{(k)} = \sum_{i=1}^n a_{ij}^{(k)}$.

We first restate the theorem of Parlett and Landis [**?**] characterizing the convergence of DPI algorithms in a slightly different form.

THEOREM A.1. *Suppose that a given scaling algorithm produces a sequence (3.2) of diagonal equivalents for a nonnegative matrix* $\mathbf{A}$ *that satisfy the following properties:*

*(P1) the sequence* $(s_k)_{k\geq 1}$ *of the product of both the row and column scaling factors*

$$s_k = \prod_{i=1}^n d_i^{(k)} e_i^{(k)}, \quad k = 1, 2, \ldots$$

*is monotonically increasing,*

*(P2) the 1-norm of rows and columns respectively converge to 1:*

$$\lim_{k\to\infty} r_i^{(k)} = 1 \text{ and } \lim_{k\to\infty} c_j^{(k)} = 1, \quad 1 \leq i, j \leq n \,,$$

*Then, if* $\mathbf{A}$ *has support,* $\widehat{\mathbf{A}} = \lim_{k\to\infty} \mathbf{A}^{(k)}$ *exists and is doubly stochastic. Additionally, if* $\mathbf{A}$ *has total support, then this limit is diagonally equivalent to* $\mathbf{A}$.

The property (P1) is exactly the same as in the hypothesis of Parlett and Landis. Property (P2) differ slightly in that it replaces and weakens partially the two other hypothesis made by Parlett and Landis, but is sufficient, combined with (P1), to raise the above conclusions (which are the same as in [**?**, Theorem 1]).

We first establish the following intermediate result.

LEMMA A.2. *Using the same notations as in Theorem A.1 above, property (P1), with the added assumption that all the elements* $a_{ij}^{(k)}$ *are bounded above by some constant independent of* $k \geq 1$, *as well as the fact that A has support, imply that*

1. $\lim_{k\to\infty} s_k = \xi > 0$ *exists,*
2. *there exists a strictly positive constant* $\gamma$ *such that*

$$d_i^{(k)} e_j^{(k)} \geq \gamma$$

*for all* $k \geq 1$ *and for each index pair* $(i, j)$ *such that* $a_{ij}$ *is strictly positive and can be permuted into a positive diagonal.*

*Proof.* Since $\mathbf{A}$ has support, there exists a permutation $\sigma$ such that $a_{i,\sigma(i)} > 0$, $1 \leq i \leq n$. Let $a = \min_{1\leq i\leq n} (a_{i,\sigma(i)})$. Then, for all $k \geq 1$,

$$\sum_{i=1}^n d_i^{(k)} e_{\sigma(i)}^{(k)} a \leq \sum_{i=1}^n d_i^{(k)} e_{\sigma(i)}^{(k)} a_{i,\sigma(i)} = \sum_{i=1}^n a_{i,\sigma(i)}^{(k)} \leq \beta \,,$$

for some constant $\beta > 0$ (the last inequality resulting from the fact that all the elements are bounded above). Then, by the arithmetic-geometric mean inequality,

$$\forall k \geq 1, \quad s_k = \prod_{i=1}^n \left( d_i^{(k)} e_{\sigma(i)}^{(k)} \right) \leq \left( \frac{1}{n} \sum_{i=1}^n d_i^{(k)} e_{\sigma(i)}^{(k)} \right)^n \leq \left( \frac{\beta}{n\, a} \right)^n \,,$$

and, combined with (P1), the monotonically increasing sequence $(s_k)_{k\geq 1}$ is thus bounded. Consequently,

$$\lim_{k\to\infty} s_k = \xi > 0$$

exists and is finite.

The demonstration of the second point goes around the same ideas. For any $a_{ij} > 0$ that can be permuted into a positive diagonal, there exists a permutation $\sigma$ such that $j = \sigma(i)$ and $a_{l,\sigma(l)} > 0$, for $1 \leq l \leq n$. Then, by (P1):

$$\forall k \geq 1, \;\; d_i^{(k)} e_j^{(k)} \prod_{\ell=1, \ell \neq i}^{n} \left( d_\ell^{(k)} e_{\sigma(\ell)}^{(k)} \right) = s_k \geq s_1 \,,$$

and

$$\forall k \geq 1, \;\; d_i^{(k)} e_j^{(k)} \geq s_1 \prod_{\ell=1, \ell \neq i}^{n} \left( d_\ell^{(k)} e_{\sigma(\ell)}^{(k)} \right)^{-1} .$$

Let $a = \min_{1 \leq i,j \leq n} (a_{ij} > 0)$. Then, for all $k \geq 1$,

$$\sum_{\ell=1, \ell \neq i}^{n} d_\ell^{(k)} e_{\sigma(\ell)}^{(k)} a \leq \sum_{\ell=1, \ell \neq i}^{n} d_\ell^{(k)} e_{\sigma(\ell)}^{(k)} a_{\ell,\sigma(\ell)} = \sum_{\ell=1, \ell \neq i}^{n} a_{\ell,\sigma(\ell)}^{(k)} \leq \beta \,,$$

and, by the arithmetic-geometric mean inequality, we can conclude that $\forall k \geq 1, \;\; d_i^{(k)} e_j^{(k)} \geq s_1 \left( \frac{(n-1)a}{\beta} \right)^{n-1} = \gamma > 0$ . □

We are now ready to prove Theorem A.1.

*Proof.* [Theorem A.1] As a direct consequence from (P2), the sequence of matrices $(\mathbf{A}^{(k)})_{k \geq 1}$ is bounded in the finite dimensional space of real, $n \times n$ matrices. Consider any convergent subsequence $(\mathbf{A}^{(q)})_q$ of $(\mathbf{A}^{(k)})_{k \geq 1}$, and define

$$\widehat{\mathbf{A}} = (\hat{a}_{ij}) = \lim_{q \to \infty} \mathbf{A}^{(q)} \,.$$

From (P2),

$$\lim_{q \to \infty} r_i^{(q)} = 1 \;\text{ and }\; \lim_{q \to \infty} c_j^{(q)} = 1, \;\; 1 \leq i, j \leq n \,,$$

which implies that $\widehat{\mathbf{A}}$ is doubly stochastic. Then, since the set of $n \times n$ doubly stochastic matrices is the convex hull of the set of $n \times n$ permutation matrices (see [**?**]), $\widehat{\mathbf{A}}$ must thus have total support. Therefore, $\hat{a}_{ij} = \lim_{q \to \infty} a_{ij}^{(q)} = 0$ whenever $a_{ij} > 0$ cannot be permuted onto a positive diagonal. Consider any entry $a_{ij} > 0$ in $\mathbf{A}$ that can be permuted onto a positive diagonal, and let

$$\mu_{ij} = \frac{\hat{a}_{ij}}{a_{ij}} = \lim_{q \to \infty} d_i^{(q)} e_j^{(q)} \,.$$

From point 2 in Lemma A.2, we know that $\mu_{ij} \geq \gamma > 0$.

Then, applying Lemma 2 of [**?**] (which is itself paraphrased from [**?**, page 345]), we know that there exist positive sequences $(x_i^{(q)})_q$ and $(y_i^{(q)})_q$ both with positive limits such that

$$d_i^{(q)} e_j^{(q)} = x_i^{(q)} y_j^{(q)}, \quad \text{for all } \hat{a}_{ij} > 0 \text{ in } \widehat{\mathbf{A}}, \;\text{ and for all } q \geq 1 \,.$$

Then, taking

$$\begin{aligned} \mathbf{X}^{(q)} &= \operatorname{diag}\left( x_1^{(q)}, \ldots, x_n^{(q)} \right) \,, \\ \mathbf{Y}^{(q)} &= \operatorname{diag}\left( y_1^{(q)}, \ldots, y_n^{(q)} \right) \,, \\ \mathbf{X} = \lim_{q \to \infty} \mathbf{X}^{(q)} \;&\text{ and }\; \mathbf{Y} = \lim_{q \to \infty} \mathbf{Y}^{(q)} \,, \end{aligned}$$

we have

$$\widehat{\mathbf{A}} = \lim_{q\to\infty} \mathbf{D}^{(q)}\mathbf{S}\mathbf{E}^{(q)} = \lim_{q\to\infty} \mathbf{X}^{(q)}\mathbf{S}\mathbf{Y}^{(q)} = \mathbf{X}\mathbf{S}\mathbf{Y} \ ,$$

showing that $\widehat{\mathbf{A}}$ is diagonally equivalent to $\mathbf{S}$, where $\mathbf{S}$ is the submatrix with total support extracted from $\mathbf{A}$ where all entries $a_{ij} > 0$ in $\mathbf{A}$ that cannot be permuted into a positive diagonal have been set to zero, the others remaining the same. Finally, if we consider any other convergent subsequence in $(\mathbf{A}^{(k)})_{k\geq 1}$, for the same reasons as above its limit will also be doubly stochastic and diagonally equivalent to $\mathbf{S}$. Since doubly stochastic equivalents are unique (see [**?**]), the two limits must be the same. Therefore, we can conclude that $\lim_{k\to\infty} \mathbf{A}^{(k)}$ exists and is doubly stochastic. If, additionally $\mathbf{A}$ has total support, then $\mathbf{A} = \mathbf{S}$ and is directly diagonally equivalent to the doubly stochastic matrix $\widehat{\mathbf{A}}$, which completes the proof of Theorem A.1. □

The previous two properties (P1) and (P2) are insufficient to prove that the two sequences of scaling matrices $\mathbf{D}^{(k)}$ and $\mathbf{E}^{(k)}$ actually converge. The reason why this additional result holds for the proposed Algorithm 1 is that, not only is the sequence $(s_k)_{k\geq 1}$ monotonically increasing, but also both sequences $(\prod_{i=1}^{n} d_i^{(k)})_{k\geq 1}$ and $(\prod_{i=1}^{n} e_i^{(k)})_{k\geq 1}$ are independently monotonically increasing. We now extend the conclusions in Theorem A.1 by incorporating this property (denoted by (P1*) in Section 3.2 and repeated below). Note that (P1*) implies property (P1).

THEOREM A.3. *Suppose that a given scaling algorithm produces a sequence* (3.2) *of diagonal equivalents for a nonnegative matrix* $\mathbf{A}$*, each satisfying the following properties:*

*(P1*) the two products of row and column scaling factors*

$$\prod_{i=1}^{n} d_i^{(k)} \quad \textit{and} \quad \prod_{i=1}^{n} e_i^{(k)}, \ k = 1, 2, \ldots$$

*are monotonically increasing*

*(P2) being the same as in Theorem A.1 above.*

*Then, if* $\mathbf{A}$ *is fully indecomposable, both limits* $\mathbf{D} = \lim_{k\to\infty} \mathbf{D}^{(k)}$ *and* $\mathbf{E} = \lim_{k\to\infty} \mathbf{E}^{(k)}$ *do exist and* $\widehat{\mathbf{A}} = \mathbf{D}\mathbf{A}\mathbf{E}$ *is doubly stochastic.*

*Proof.* Since (P1*) implies (P1), and since (P2) remains the same, we know from Theorem A.1 that if $\mathbf{A}$ has total support, $\widehat{\mathbf{A}} = \lim_{k\to\infty} \mathbf{A}^{(k)}$ exists and is diagonally equivalent to $\mathbf{A}$. To prove that both $\mathbf{D} = \lim_{k\to\infty} \mathbf{D}^{(k)}$ and $\mathbf{E} = \lim_{k\to\infty} \mathbf{E}^{(k)}$ exist and $\widehat{\mathbf{A}} = \mathbf{D}\mathbf{A}\mathbf{E}$, we exploit the assumption that $\mathbf{A}$ is fully indecomposable. In this case, $\mathbf{A}$ is diagonally equivalent to the doubly stochastic limit $\widehat{\mathbf{A}} = \lim_{k\to\infty} \mathbf{A}^{(k)}$, and the diagonal matrices which take place in this equivalence are unique up to a scalar factor [**?**].

Let us suppose now that one of the sequences $(d_i^{(k)})_{k\geq 1}$ is unbounded for some $i$. In such a case, there exist a subsequence $(d_i^{(q)})_q$ such that

$$\lim_{q\to\infty} d_i^{(q)} = +\infty \ .$$

As the matrix $\mathbf{A}$ is fully indecomposable, for any index $j$, $1 \leq j \leq n$, there exist a chain of positive entries in which the row and column indexes alternately change

$$a_{ij_1}, a_{i_1j_1}, a_{i_1j_2}, a_{i_2j_2}, a_{i_2j_3}, \ldots, a_{i_{p-1}j_{p-1}}, a_{i_{p-1}j_p}, a_{i_pj_p}, a_{i_pj}$$

"*connecting*" row $i$ to column $j$ (see, for example, [**?**, Theorem 4.2.7]). Consequently, because of the conclusions raised at point 2 in Lemma A.2, we know that the denominator in the following fraction

$$\frac{d_i^{(k)} e_{j_1}^{(k)}\, d_{i_1}^{(k)} e_{j_2}^{(k)}\, d_{i_2}^{(k)} e_{j_3}^{(k)} \;\ldots\; d_{i_{p-2}}^{(k)} e_{j_{p-1}}^{(k)}\, d_{i_{p-1}}^{(k)} e_{j_p}^{(k)}\, d_{i_p}^{(k)} e_j^{(k)}}{d_{i_1}^{(k)} e_{j_1}^{(k)}\, d_{i_2}^{(k)} e_{j_2}^{(k)} \;\ldots\; d_{i_{p-1}}^{(k)} e_{j_{p-1}}^{(k)}\, d_{i_p}^{(k)} e_{j_p}^{(k)}} \;=\; d_i^{(k)} e_j^{(k)}$$

is bounded away from zero and. From the demonstration of Theorem A.1 above this fraction has a strictly positive limit

$$\lim_{k\to\infty} d_i^{(k)} e_j^{(k)} \;=\; \frac{\mu_{ij_1}\,\mu_{i_1j_2}\,\mu_{i_2j_3}\;\cdots\;\mu_{i_{p-2}j_{p-1}}\,\mu_{i_{p-1}j_p}\,\mu_{i_pj}}{\mu_{i_1j_1}\,\mu_{i_2j_2}\;\cdots\;\mu_{i_{p-1}j_{p-1}}\,\mu_{i_pj_p}} \;=\; \mu_{ij} > 0\,.$$

Since this can be done for any $j$, we can conclude that the subsequence $(\prod_{j=1}^n e_j^{(q)})_q$ (which is strictly positive) goes to zero as $(d_i^{(q)})^{-n}\prod_{j=1}^n \mu_{ij}$. This last conclusion is in contradiction with property (P1*) above, stating that both sequences $(\prod_{j=1}^n d_j^{(k)})_{k\geq 1}$ and $(\prod_{j=1}^n e_j^{(k)})_{k\geq 1}$ are monotonically increasing. The same could be done with one of the $(e_j^{(k)})_{k\geq 1}$ instead, and we can finally conclude that each of the sequences $(d_i^{(k)})_{k\geq 1}$ and $(e_i^{(k)})_{k\geq 1}$ are bounded.

Now, since the two sequences $(\mathbf{D}^{(k)})_{k\geq 1}$ and $(\mathbf{E}^{(k)})_{k\geq 1}$ are bounded in the finite dimensional space of real $n\times n$ diagonal matrices, they have convergent subsequences. Let us consider two convergent subsequences:

$$\left(\mathbf{D}^{(p)}, \mathbf{E}^{(p)}\right) \underset{p\to+\infty}{\longrightarrow} (\mathbf{D}_1, \mathbf{E}_1) \;\text{ and }\; \left(\mathbf{D}^{(q)}, \mathbf{E}^{(q)}\right) \underset{q\to+\infty}{\longrightarrow} (\mathbf{D}_2, \mathbf{E}_2)\,.$$

Obviously, $\mathbf{D}_1\mathbf{A}\mathbf{E}_1 = \widehat{\mathbf{A}} = \mathbf{D}_2\mathbf{A}\mathbf{E}_2$, and thus, because of the uniqueness (shown in [**?**]), there exists $\alpha > 0$ such that $\mathbf{D}_1 = \alpha\mathbf{D}_2$ and $\mathbf{E}_1 = (1/\alpha)\mathbf{E}_2$. Then, as mentioned above, since both sequences $\left(\prod_{i=1}^n d_i^{(k)}\right)_{k\geq 1}$ and $\left(\prod_{i=1}^n e_i^{(k)}\right)_{k\geq 1}$ are monotonically increasing, it is clear that $\alpha$ must be equal to 1 and the two limits must be equal. Therefore, we can conclude that $\mathbf{D} = \lim_{k\to\infty}\mathbf{D}^{(k)}$ and $\mathbf{E} = \lim_{k\to\infty}\mathbf{E}^{(k)}$ exist, which completes the proof. □

The matrices with total support can be permuted into a direct sum of fully indecomposable matrices [**?**, Theorem 1]. Now, if the algorithm generates iterates (3.2) that are insensitive to permutations of $\mathbf{A}$, and if applying it to a block diagonal matrix is equivalent to working directly which each diagonal block separately, we can easily extend the conclusions of Theorem A.3 to the case of matrices with total support by collecting the conclusions from Theorem A.3 independently on each fully indecomposable sub-matrix in such a direct sum. This is presented as the following corollary.

COROLLARY A.4. *Suppose that a given scaling algorithm produces a sequence of diagonal equivalents, as in (3.2), for a nonnegative matrix* $\mathbf{A}$ *that satisfy the three properties (P1*) and (P2), given in Theorem A.3 above, and if additionally*

1. *the algorithm is permutation insensitive, in the sense that, under any row or column permutation of the original matrix* $\mathbf{A}$*, the scaling elements remain the same and are just permuted along the diagonals of* $\mathbf{D}^{(k)}$ *and* $\mathbf{E}^{(k)}$ *with respect to the corresponding row or column permutations applied to* $\mathbf{A}$*,*
2. *the iterates also remain the same when applying the algorithm to a block diagonal matrix as when collecting results obtained by applying the algorithm on each diagonal submatrix separately,*

*then, if* $\mathbf{A}$ *has total support, both limits* $\mathbf{D} = \lim_{k\to\infty} \mathbf{D}^{(k)}$ *and* $\mathbf{E} = \lim_{k\to\infty} \mathbf{E}^{(k)}$ *do exist and* $\widehat{\mathbf{A}} = \mathbf{DAE}$ *is doubly stochastic.*

The two extra hypothesis of Corollary A.4 are naturally satisfied by algorithms that generate iterates by relying only on the computation of the one norm of both rows and columns at each iteration (as is the case in the proposed and Sinkhorn-Knopp algorithms). We recall that Parlett and Landis use the fact that their algorithm incorporates a normalization step, with

$$\mu_k = \frac{1}{n} \sum_{i=1}^{n} r_i^{(k+1)} = 1 \ ,$$

enforcing the mean of all row sums to be set to 1 at each iteration. This was actually incorporated as property (P3) in the hypothesis of their main convergence theorem. As we have seen above, this is not mandatory to reach the same conclusions as in Theorem A.1. Additionally, this normalization step may however violate the second hypothesis in Corollary A.4, because the averaging of row sums may not be the same when computed globally or block-wise in the case of a block-diagonal matrix. This may not prevent anyway the algorithms proposed in [**?**] (they incorporate this normalization step) to produce still a convergent sequence of scaling factors. We have not found how to characterize in a simple way some common generic properties, that might enable to reach the same conclusions in the case of matrices with total support only, and extend the scope of Corollary A.4 to address these particular "*normalized*" scaling algorithms.