Acta Mathematica Academiae Paedagogicae Nyíregyháziensis
25 (2009), 283–301
www.emis.de/journals
ISSN 1786-0091

SCALING PROBLEMS IN LINEAR-FRACTIONAL PROGRAMMING

E. BAJALINOV AND A. RÁCZ

Abstract. In this paper we discuss the theoretical backgrounds and implementation issues of scaling a linear-fractional programming problem (LFP). We consider an LFP problem in the canonical form and show how to scale rows and columns of the problem, then if the scaled problem has been solved, we show how the solution obtained may be *un-scaled*. We also overview briefly three scaling rules for calculating scaling factors. Finally, to illustrate how these rules work we consider a numeric example.

1. Introduction

Let us consider the following canonical LFP problem:

$$Q(x) = \frac{P(x)}{D(x)} \to \max \tag{1}$$

subject to

$$\sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, 2, \dots, m, \tag{2}$$

$$x_j \geq 0, \quad j = 1, 2, \dots, n, \tag{3}$$

where

$$P(x) = \sum_{j=1}^{n} p_j x_j + p_0, \qquad D(x) = \sum_{j=1}^{n} d_j x_j + d_0,$$

$D(x) > 0, \quad \forall x = (x_1, x_2, \dots, x_n)^T \in S, \quad S$ - is a feasible set defined by constraints (2) and (3).

It is a well-known fact that large linear-programming (LP) models

$$P(x) = \sum_{j=1}^{n} p_j x_j \to \max \tag{4}$$

2000 *Mathematics Subject Classification.* 90C32.
Key words and phrases. linear-fractional programming, scaling.

subject to constraints (2)-(3); require hundreds of thousands to millions of floating-point arithmetic calculations to solve. Because of the finite precision inherent in computer arithmetic, small numerical errors occur in these calculations. These errors typically have a cumulative effect, that often leads to a numerically unstable problem and possibly large errors in the 'solution' obtained. The same computational problems occur in large-scale LFP problems too.

To avoid such problems, all well-made industrial LP solvers include special sophisticated techniques, that dramatically reduce the cumulative effect of rounding and often lead to considerable improvement in the solvers' performance.

One of the most easy, relatively effective and widespread techniques of this type is *scaling*[1]. This technique means that those rows and/or columns of matrix $A = \|a_{ij}\|_{m\times n}$ in the original optimization problem, which are *poorly* (or *badly*) scaled, that is have a wide range of entries, must be divided (or multiplied) with their own scaling factors $\rho_i^r,\ i = 1, 2, \ldots, m$ and/or $\rho_j^c,\ j = 1, 2, \ldots, n,$ respectively. In most real-world LP and LFP applications the model originally is very poorly scaled – for example, with dollar amounts in millions for some constraints and return figures in percentages for others. This is why before beginning the simplex or other method the program package must scale columns, rows, and right-hand sides to a common magnitude.

Such scaling may include or may not the coefficients of the objective function. In the case of LP problems, scaling matrix A, right-hand-side vector b and objective function $P(x)$ does not lead to any difficulties because of the linearity of the constraints and the objective function. In the most cases scaling improves the numerical properties of the problem to be solved so it is justified to use it. Moreover, sometimes it can dramatically reduce the number of iterations in simplex method. Most professionally developed LP solvers automatically use scaling methods to maintain numerical stability and improve performance.

Normally, you can choose among such options as: 'No Scaling', 'Row Scaling', 'Column Scaling', or 'Row and Column Scaling' with or without scaling the objective function.

In the case of LFP problem (1)-(3), when scaling we should keep in mind the main difference between LP and LFP problems – the non-linear objective function $Q(x)$.

In this paper we consider the theoretical backgrounds of such type of techniques that are usually used to make solvers more stable and can help to improve their performance.

---

[1]The pre-solution transformation of the data of a problem that attempts to make the magnitudes of all the data as close as possible.

2. Scaling problems and un-scaling solutions

When scaling LFP problem, we have to distinguish the following cases:

(1) scaling constraints:
- right-hand side vector $b = (b_1, b_2, \dots, b_m)^T$;
- columns A_j, $j = 1, 2, \dots, n$; of matrix A;
- rows of matrix A;

(2) scaling objective function:
- only vector $p = (p_0, p_1, p_2, \dots, p_n)$ of numerator $P(x)$;
- only vector $d = (d_0, d_1, d_2, \dots, d_n)$ of denominator $D(x)$;
- both vectors p and d of objective function $Q(x)$.

Below, we investigate all these possible cases.

2.1. **Right-hand side column-vector.** Suppose that vector x^* is an optimal solution for LFP problem (1)-(3), so

$$\sum_{j=1}^{n} A_j x_j^* = b \quad \text{and} \quad x^* \geq 0,$$

and matrix $B = (A_{s_1}, A_{s_2}, \dots, A_{s_m})$ is its basis.

Let us replace RHS vector b with some other vector $b' = \rho\, b$, where $\rho > 0$. Consider the new vector $x' = \rho x^*$. It is obvious that this vector x' satisfies constraints

$$\sum_{j=1}^{n} A_j (\rho x_j^*) = \rho\, b \quad \text{and} \quad x' = \rho x^* \geq 0,$$

so vector x' is a feasible solution of LFP problem

$$Q(x) = \frac{P(x)}{D(x)} = \frac{\sum_{j=1}^{n} p_j x_j + p'_0}{\sum_{j=1}^{n} d_j x_j + d'_0} \to \max \tag{5}$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j = \rho b_i, \quad i = 1, 2, \dots, m; \tag{6}$$

$$x_j \geq 0, \quad j = 1, 2, \dots, n. \tag{7}$$

Now we have to check this vector x' whether it is an optimal solution of problem (5)-(7). Since vector x^* is an optimal solution of the original LFP problem (1)-(3), we have

$$\Delta_j(x^*) \geq 0, \;\; j = 1, 2, \dots, n, \tag{8}$$

where

$$\Delta_j(x^*) = D(x^*)\Delta'_j - P(x^*)\Delta''_j, \quad j = 1, 2, \ldots, n,$$

$$\Delta'_j = \sum_{i=1}^{m} p_{s_i} x_{ij} - p_j, \quad j = 1, 2, \ldots, n,$$

$$\Delta''_j = \sum_{i=1}^{m} d_{s_i} x_{ij} - d_j, \quad j = 1, 2, \ldots, n,$$

coefficient x_{ij} are defined from the systems

$$\sum_{i=1}^{m} A_{s_i} x_{ij} = A_j, \quad j = 1, 2, \ldots, n; \tag{9}$$

and A_j denotes the column-vectors $A_j = (a_{1j}, a_{2j}, \ldots, a_{mj})^T$, $j = 1, 2, \ldots, n$, of matrix $A = \|a_{ij}\|_{m\times n}$.

Observe that reduced costs Δ'_j and Δ''_j do not depend on RHS vector b, so substitution $b \to \rho\, b$ does not affect values of Δ'_j and Δ''_j. However, values of functions $P(x)$ and $D(x)$ depend on RHS vector b, so we have to consider the new reduced costs $\Delta_j(x')$, where $x' = \rho x^*$, for LFP problem (5)-(7). We have

$$\begin{aligned}
\Delta_j(\rho x^*) &= D(\rho x^*)\ \Delta'_j - P(\rho x^*)\ \Delta''_j = \\
&= (\sum_{j=1}^{n} d_j(\rho x^*_j) + d'_0)\ \Delta'_j - (\sum_{j=1}^{n} p_j(\rho x^*_j) + p'_0)\ \Delta''_j = \\
&= (\sum_{j=1}^{n} d_j(\rho x^*_j) + d'_0 + \rho d_0 - \rho d_0)\ \Delta'_j - \\
&\quad -(\sum_{j=1}^{n} p_j(\rho x^*_j) + p'_0 + \rho p_0 - \rho p_0)\ \Delta''_j = \\
&= \rho D(x^*)\ \Delta'_j + (d'_0 - \rho d_0)\ \Delta'_j - \rho P(x^*)\ \Delta''_j - (p'_0 - \rho p_0)\ \Delta''_j = \\
&= \rho \Delta_j(x^*) + (d'_0 - \rho d_0)\ \Delta'_j - (p'_0 - \rho p_0)\ \Delta''_j = \\
&= \rho \Delta_j(x^*) - G_j, \qquad (10)
\end{aligned}$$

where

$$G_j = (p'_0 - \rho p_0)\ \Delta''_j - (d'_0 - \rho d_0)\ \Delta'_j\ .$$

Formula (10) means that if p'_0 and d'_0 are such that

$$\rho \Delta_j(x^*) - G_j \geq 0, \quad j = 1, 2, \ldots, n,$$

or, in particular, if $p'_0 = \rho p_0$ and $d'_0 = \rho d_0$, then

$$\Delta_j(\rho x^*) \overset{(10)}{=} \rho \Delta_j(x^*) \overset{(8)}{\geq} 0, \quad \forall j = 1, 2, \ldots, n,$$

and hence, vector x' is an optimal solution of LFP problem (5)-(7).

So, if we substitute RHS vector b with some other vector $b' = \rho b,\ \rho > 0,$ we have simultaneously to replace coefficients p_0 and d_0 in the original objective function $Q(x)$ with $p'_0 = \rho p_0$ and $d'_0 = \rho d_0$, respectively. These two substitutions will guarantee the equivalence between the original problem (1)-(3) and the new scaled LFP problem (5)-(7).

It is obvious that if vector x' is an optimal solution of the new (scaled) LFP problem (5)-(7), then vector $x^* = x'/\rho$ is an optimal solution of the original LFP problem (1)-(3).

2.2. **Left-hand side column-vectors.** In this section we consider the scaling columns $A_j,\ j = 1, 2, \dots, n$, of matrix $A = \|a_{ij}\|_{m\times n}$. We suppose that vector x^* is an optimal solution for the original LFP problem (1)-(3), so

$$\sum_{j=1}^{n} A_j\, x_j^* = b \quad \text{and} \quad x_j^* \geq 0,\ j = 1, 2, \dots, n,$$

and matrix $B = (A_{s_1}, A_{s_2}, \dots, A_{s_m})$ is its basis.

Let us replace vector $A_r,\ r \in J = \{1, 2, \dots, n\}$, with some other vector $A'_r = \rho A_r$, where $\rho > 0$. It is obvious that new vector

$$x' = (x_1^*,\ x_2^*,\ \dots,\ x_{r-1}^*,\ \frac{x_r^*}{\rho},\ x_{r+1}^*,\ \dots,\ x_n^*)$$

will satisfy constraints

$$\sum_{\substack{j=1\\ j\neq r}}^{n} A_j x_j^* + \rho A_r \frac{x_r^*}{\rho} = b,$$

$$x'_j \geq 0, \quad j = 1, 2, \dots, n,$$

and hence vector x' is a feasible solution of the new scaled LFP problem

$$Q(x) = \frac{P'(x)}{D'(x)} = \frac{\sum\limits_{\substack{j=1\\ j\neq r}}^{n} p_j x_j + p'_r x_r + p_0}{\sum\limits_{\substack{j=1\\ j\neq r}}^{n} d_j x_j + d'_r x_r + d_0} \to \max \tag{11}$$

subject to

$$\sum_{\substack{j=1\\ j\neq r}}^{n} A_j x_j + A'_r x_r = b, \tag{12}$$

$$x_j \geq 0, \quad j = 1, 2, \dots, n. \tag{13}$$

Our aim is now to detect whether vector x' is an optimal solution of the scaled LFP problem (11)-(13)? Since vector x^* is an optimal solution of the original problem (1)-(3), we have that

$$\Delta_j(x^*) \;=\; D(x^*)\Delta'_j - P(x^*)\Delta''_j \;\geq 0, \quad j = 1, 2, \dots, n. \tag{14}$$

Here we have to distinguish the following two cases: A_r is a basic vector, and vector A_r is a non-basic vector.

Let us suppose that A_r is a basic vector, i.e. $r \in J_B = \{s_1, s_2, \dots, s_m\}$. In this case, keeping in mind that

$$P'(x') = \sum_{\substack{j=1 \\ j \neq r}}^{n} p_j x^*_j + p'_r \frac{x^*_r}{\rho} + p_0\,, \qquad D'(x') = \sum_{\substack{j=1 \\ j \neq r}}^{n} d_j x^*_j + d'_r \frac{x^*_r}{\rho} + d_0\,,$$

$$\Delta'_j = \sum_{\substack{i=1 \\ s_i \neq r}}^{m} p_{s_i} x_{ij} + p'_r \frac{x_{rj}}{\rho} - p_j\,, \qquad \Delta''_j = \sum_{\substack{i=1 \\ s_i \neq r}}^{m} d_{s_i} x_{ij} + d'_r \frac{x_{rj}}{\rho} - d_j\,.$$

for the new scaled problem (11)-(13) we have

$$\begin{aligned} \Delta_j(x') &= D'(x')\Delta'_j - P'(x')\Delta''_j = \\ &= (D(x^*) - d_r x^*_r + d'_r \frac{x^*_r}{\rho})(\Delta'_j - p_r x_{rj} + p'_r \frac{x_{rj}}{\rho}) - \\ &- (P(x^*) - p_r x^*_r + p'_r \frac{x^*_r}{\rho})(\Delta''_j - d_r x_{rj} + d'_r \frac{x_{rj}}{\rho}) \end{aligned} \tag{15}$$

Equality (15) makes it obvious that if $p'_r = p_r\rho$ and $d'_r = d_r\rho$, then

$$\Delta_j(x') \overset{(15)}{=} \Delta_j(x^*) \overset{(14)}{\geq} 0, \quad j = 1, 2, \dots, n.$$

The latter means that in this case vector x' is an optimal solution of the scaled LFP problem (11)-(13). So, if we substitute some basic vector A_r with some other vector $A'_r = \rho\, A_r$, $\rho > 0$, we simultaneously have to replace coefficients p_r and d_r in the original objective function $Q(x)$ with $p'_r = \rho p_r$ and $d'_r = \rho d_r$, respectively. These two substitutions will guarantee the equivalence between the original problem (1)-(3) and the new scaled LFP problem (11)-(13).

It is obvious that if vector x' is an optimal solution of the new (scaled) LFP problem (11)-(13), then vector $x^* = (x'_1,\, x'_2,\, \dots,\, x'_{r-1},\, x'_r\rho,\, x'_{r+1},\, \dots,\, x'_n)$ will be an optimal solution of the original LFP problem (1)-(3).

Now, we have to consider the case when substituted vector A_r is a non-basic vector, i.e. $r \in J_N = J \setminus J_B$. As in the previous case, we simultaneously replace original coefficients p_r and d_r with ρp_r and ρd_r, respectively. Since index r is non-basic and $x^*_r = 0$, it is obvious that

$$x' = x^*, \;\; P'(x') = P(x^*), \;\; D'(x') = D(x^*) \quad \text{and hence} \quad Q'(x') = Q(x^*).$$

So replacement $A_r \;\rightarrow\; \rho A_r$, $r \in J_N$, affects only values of Δ'_r, Δ''_r, and $\Delta_r(x')$. Indeed, if in the original LFP problem (1)-(3) for non-basic vector A_r

we had (see (9)), that

$$\sum_{i=1}^{m} A_{s_i} x_{ir} = A_r, \quad r \in J_N,$$

then after replacement $A_r \to A'_r$, where $A'_r = \rho A_r$, we obtain the following representation of the new vector A'_r in the same basis B:

$$\sum_{i=1}^{m} A_{s_i} (\rho x_{ir}) = \rho A_r, \quad r \in J_N.$$

If when replacing $A_r \to \rho A_r$, we simultaneously substitute $p_r \to p'_r$, where $p'_r = \rho p_r$, and $d_r \to d'_r$, where $d'_r = \rho d_r$, then for new $\tilde{\Delta}'_r$, $\tilde{\Delta}''_r$, and $\tilde{\Delta}_r(x')$ we have

$$\tilde{\Delta}'_r = \sum_{i=1}^{m} p_{s_i} \, (\rho x_{ir}) - (\rho p_r) = \rho \Delta'_r,$$

$$\tilde{\Delta}''_r = \sum_{i=1}^{m} d_{s_i} \, (\rho x_{ir}) - (\rho d_r) = \rho \Delta''_r.$$

Thus,

$$\tilde{\Delta}_r(x') = D(x^*)\tilde{\Delta}'_r - P(x^*)\tilde{\Delta}''_r = D(x^*)(\rho\Delta'_r) - P(x^*)(\rho\Delta''_r) = \rho\Delta_r(x^*) \overset{(14)}{\geq} 0.$$

The latter means that in this case vector x^* is an optimal solution of the scaled LFP problem (11)-(13).

So, if we substitute some non-basic vector A_r with some other vector $A'_r = \rho\, A_r$, $\rho > 0$, we have simultaneously to replace coefficients p_r and d_r in the original objective function $Q(x)$ with $p'_r = \rho p_r$ and $d'_r = \rho d_r$, respectively. These two substitutions will guarantee the equivalence between the original problem (1)-(3) and the new scaled LFP problem (11)-(13). Moreover, it will guarantee that $x^*_r = x'_r = 0$.

2.3. **Rows of constraints.** Let us replace row-vector $a_r = (a_{r1}, a_{r2}, \dots, a_{rn})$ of matrix $A = ||a_{ij}||_{m\times n}$ in LFP problem (1)-(3) with some other row-vector $a'_r = \rho a_r$. In this case we have to distinguish the following two cases:

(1) simultaneously with replacement $a_r \to \rho a_i$ we substitute the r-th element of RHS vector b, that is $b_r \to b'_r = \rho b_r$.

(2) we do not modify any element in RHS column-vector, so scaling must be performed only in matrix A.

In the case (1) instead of original constraint in the r-th row

$$\sum_{j=1}^{n} a_{rj} x_j = b_r, \quad \text{we have} \quad \sum_{j=1}^{n} (\rho a_{rj}) x_j = (\rho b_r).$$

It is well-known that such scaling does not affect the structure of feasible set S. So the new scaled problem is absolutely equivalent with the original one.

In case (2) we do not modify RHS vector b. Such scaling leads to unpredictable deformations in feasible set S, so we cannot provide any guarantee that the optimal basis of the scaled problem will be the same as in the original one. So, the only negotiable method of scaling rows in matrix A is the following $\tilde{a}_r \to \tilde{a}'_r$ where $\tilde{a}_r = (a_{r1}, a_{r2}, \dots, a_{rn}, b_r)$ and $\tilde{a}'_r = (\rho a_{r1}, \rho a_{r2}, \dots, \rho a_{rn}, \rho b_r)$.

Obviously, the optimal solution x' of the scaled LFP problem is absolutely identical with the optimal solution x^* of the original LFP problem. So we need not any 'un-scaling' in this case.

Note that in the simplex method, when performing θ-ratio test, the elements of the pivotal column take part in the calculations. Hence, the choice of pivotal row depends on the row scaling. Since a bad choice of pivots can lead to large errors in the computed solution, it means that a proper row scaling is very important.

2.4. **Objective function.**

2.4.1. *Numerator $P(x)$.* Let us replace vector $p = (p_0, p_1, \dots, p_n)$ in the numerator $P(x)$ with some other vector $p' = (p'_0, p'_1, \dots, p'_n)$, where $p'_j = \rho p_j$, $j = 0, 1, 2, \dots, n$.

It is clear that this replacement does not affect either the optimal value of denominator $D(x)$ or the values of reduced costs Δ''_j, $j = 1, 2, \dots, n$, but it changes the optimal values of functions $P(x)$ and $Q(x)$ and affects the values of reduced costs Δ'_j and $\Delta_j(x)$, $j = 1, 2, \dots, n$.

So, for the new values $\tilde{\Delta}'_j$, $P'(x^*)$, $Q'(x^*)$, and $\tilde{\Delta}_j(x^*)$, $j = 1, 2, \dots, n$, we have:

$$\tilde{\Delta}'_j = \sum_{i=1}^{m} p'_{s_i} x_{ij} - p'_j = \sum_{i=1}^{m} (\rho\, p_{s_i}) x_{ij} - (\rho\, p_j) = \rho\, \Delta'_j, \quad j = 1, 2, \dots, n,$$

$$P'(x^*) = \sum_{j=1}^{n} p'_j x^*_j + p'_0 = \sum_{j=1}^{n} (\rho\, p_j) x^*_j + (\rho\, p_0) = \rho\, P(x^*),$$

$$Q'(x^*) = P'(x^*)/D(x^*) = \rho\, P(x^*)/D(x^*) = \rho\, Q(x^*),$$

and hence,

$$\begin{aligned}\tilde{\Delta}_j(x^*) &= D(x^*)\tilde{\Delta}'_j - P'(x^*)\Delta''_j = \\ &= D(x^*)(\rho\, \Delta'_j) - (\rho\, P(x^*))\Delta''_j = \rho\, \Delta_j(x^*), \quad j = 1, 2, \dots, n.\end{aligned}$$

Since $\rho > 0$ and $\Delta_j(x^*) \geq 0$, $j = 1, 2, \dots, n$, the latter means that

$$\tilde{\Delta}_j(x^*) = \rho\, \Delta_j(x^*) \overset{(14)}{\geq} 0, \quad j = 1, 2, \dots, n.$$

Finally, we have to note that replacement $p \to \rho p$ does not lead to any changes in the optimal basis or in optimal solution x^*. So, if we have solved the scaled LFP problem, in order to 'un-scale' the optimal solution obtained we have to use the following formula $Q(x^*) = \dfrac{1}{\rho}\, Q'(x^*)$, because the optimal solution x' of the scaled problem is exactly the same as optimal solution x^* of the original problem.

2.4.2. *Denominator* $D(x)$. Let us replace vector $d = (d_0, d_1, \dots, d_n)$ in the denominator $D(x)$ with another vector $d' = (d'_0, d'_1, \dots, d'_n)$, where $d'_j = \rho d_j,\ j = 0, 1, \dots, n$.

It is obvious that such replacement leads to some changes in the optimal values of denominator $D(x)$, objective function $Q(x)$ and values $\Delta''_j,\ \Delta_j(x),\ j = 1, 2, \dots, n$; but does not affect the optimal value of numerator $P(x)$ or the values of reduced costs $\Delta'_j,\ j = 1, 2, \dots, n$.

So for new values $\tilde{\Delta}''_j,\ D'(x^*),\ Q'(x^*),$ and $\tilde{\Delta}_j(x^*),\ \ j = 1, 2, \dots, n,$ we have

$$\tilde{\Delta}''_j = \sum_{i=1}^{m} d'_{s_i} x_{ij} - d'_j = \sum_{i=1}^{m} (\rho\, d_{s_i}) x_{ij} - (\rho\, d_j) = \rho\, \Delta''_j, \quad j = 1, 2, \dots, n,$$

$$D'(x^*) = \sum_{j=1}^{n} d'_j x^*_j + d'_0 = \sum_{j=1}^{n} (\rho\, d_j) x^*_j + (\rho\, d_0) = \rho\, D(x^*),$$

$$Q'(x^*) = P(x^*)/D'(x^*) = P(x^*)/(\rho\, D(x^*)) = Q(x^*)/\rho,$$

and hence,

$$\begin{aligned}\tilde{\Delta}_j(x^*) &= D'(x^*)\Delta'_j - P(x^*)\tilde{\Delta}''_j = \\ &= (\rho\, D(x^*))\Delta'_j - P(x^*)(\rho\, \Delta''_j) = \rho\, \Delta_j(x^*), \quad j = 1, 2, \dots, n.\end{aligned}$$

Thus, we obtain

$$\tilde{\Delta}_j(x^*) = \rho\, \Delta_j(x^*) \overset{(14)}{\geq} 0, \quad j = 1, 2, \dots, n.$$

Finally, we have to note that replacement $d \to \rho d$ does not lead to any changes in optimal basis B or in optimal solution x^*. So once we have solved the scaled LFP problem, in order to 'un-scale' the optimal solution obtained we have to use the following formula $Q(x^*) = \rho\, Q'(x^*)$, because the optimal solution x' of the scaled problem is exactly the same as optimal solution x^* of the original problem.

3. Scaling factors and implementations issues

In this section we briefly overview three rules for calculating scaling factors ρ. The first two techniques have been implemented in several commercial and freely usable LP codes, and bring a compromise between provided stability and computational efficiency. For more information on scaling rules with detailed theoretical backgrounds see, for instance [2], [3], [4], [5], etc.

Consider a matrix $A = ||a_{ij}||_{m\times n}$ and an RHS vector $b = (b_1, b_2, \dots, b_m)^T$. A measure of *ill-scaling* of the system $Ax = b$ is

$$\sigma(A) \;=\; \max_{i,j\in J_+}(|a_{ij}|) \;/\; \min_{i,j\in J_+}(|a_{ij}|),$$

where $J_+ = \{i, j \mid a_{ij} \neq 0\}$. The larger is the magnitude between the largest and the smallest absolute values of non-zero entries a_{ij}, the worse scaled is the system.

We will say that given matrix A is **poorly scaled** or **badly scaled**, if $\sigma(A) >= 1E + 5$.

The aim of scaling is to make measure $\sigma(A)$ as small as possible. To reach this aim we can scale columns and rows as many times as we need.

3.1. **Geometric rule.** In accordance with this rule we define the following column-vector ρ^r of scaling factors for rows

$$\rho^r \;=\; (\rho_1^r, \rho_2^r, \dots, \rho_m^r)^T, \tag{16}$$

where

$$\rho_i^r \;=\; (\prod_{j\in J_i^+} a_{ij})^{1/K_i^r}, \quad i = 1, 2, \dots, m;$$

$J_i^+ = \{j : a_{ij} \neq 0\}$, $i = 1, 2, \dots, m$, is a row related set of indices j of non-zero entries a_{ij} in row i, and K_i^r denotes the number of non-zero entries a_{ij} in row i.

Analogically, to scale columns, we use the following factors organized in a row-vector ρ^c

$$\rho^c \;=\; (\rho_1^c, \rho_2^c, \dots, \rho_n^c), \tag{17}$$

where

$$\rho_j^c \;=\; (\prod_{i\in I_j^+} a_{ij})^{1/K_j^c}, \quad j = 1, 2, \dots, n;$$

$I_j^+ = \{i : a_{ij} \neq 0\}$, $j = 1, 2, \dots, n$, is a column related set of indices i of non-zero entries a_{ij} in column j, and K_j^c denotes the number of non-zero entries a_{ij} in column j.

3.2. **Mean rule.** As an alternative to the scaling factors calculated in accordance with Geometric rule, we can define the following column-vector ρ^r of scaling factors for rows

$$\rho^r \;=\; (\rho_1^r, \rho_2^r, \dots, \rho_m^r)^T, \tag{18}$$

where

$$\rho_i^r \;=\; \sqrt{r_i'\, r_i''}, \quad i = 1, 2, \dots, m;$$

and

$$r_i' = \max_{j:\, a_{ij}\neq 0} |a_{ij}|, \quad r_i'' = \min_{j:\, a_{ij}\neq 0} |a_{ij}|, \quad i = 1, 2, \dots, m.$$

Analogically to row scaling factors, for columns we have to define the following row-vector ρ^c of scaling factors

$$\rho^c = (\rho^c_1, \rho^c_2, \ldots, \rho^c_n), \tag{19}$$

where

$$\rho^c_j = \sqrt{c'_j\, c''_j}, \quad j = 1, 2, \ldots, n;$$

and

$$c'_j = \max_{i:\, a_{ij} \neq 0} |a_{ij}|, \quad c''_j = \min_{i:\, a_{ij} \neq 0} |a_{ij}|, \quad j = 1, 2, \ldots, n.$$

3.3. **Max/Min-rule.** As an alternative to the previous two scaling rules, we consider the following method for calculating scaling factors. First, we define the smallest $a_{i_0 j_0}$ and the largest $a_{i_1 j_1}$ absolute values for non-zero entries in the matrix and then calculate the following four scaling factors:

ρ'_c – for the column j_0 containing the minimal value,
ρ''_c – for the column j_1 containing the maximal value,
ρ'_r – for the row i_0 containing the minimal value,
ρ''_r – for the row i_1 containing the maximal value,

using the following formulas:

$$\rho'_c = \frac{\max\limits_i |a_{ij_0}| + a_{i_1 j_1}}{2 \times \max\limits_i |a_{ij_0}|}, \qquad \rho''_c = \frac{\min\limits_{i:\, a_{ij} \neq 0} |a_{ij_1}| + a_{i_0 j_0}}{2 \times \min\limits_{i:\, a_{ij} \neq 0} |a_{ij_1}|},$$

$$\rho'_r = \frac{\max\limits_j |a_{i_0 j}| + a_{i_1 j_1}}{2 \times \max\limits_j |a_{i_0 j}|}, \qquad \rho''_r = \frac{\min\limits_{j:\, a_{ij} \neq 0} |a_{i_1 j}| + a_{i_0 j_0}}{2 \times \min\limits_{j:\, a_{ij} \neq 0} |a_{i_1 j}|}.$$

It is obvious that when applying such scaling factors the 'distance' between the largest and the smallest absolute values in the matrix decreases.

3.4. **Implementation Issues.** To scale an LFP problem we have to calculate and then to store scaling factors for rows, columns and the objective function (separately for numerator and denominator). One of the possible ways to store factors is to expand the matrix of the problem as follows

$$\tilde{A} = \left(\begin{array}{c|cccc|c}
\rho^r_1 & a_{11} & a_{12} & \ldots & a_{1n} & b_1 \\
\rho^r_2 & a_{21} & a_{22} & \ldots & a_{2n} & b_2 \\
\vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\
\rho^r_m & a_{m1} & a_{m2} & \ldots & a_{mn} & b_n \\
\hline
\rho^r_{m+1} & p_1 & p_2 & \ldots & p_n & p_0 \\
\rho^r_{m+2} & d_1 & d_2 & \ldots & d_n & d_0 \\
\hline
 & \rho^c_1 & \rho^c_2 & \ldots & \rho^c_n & \rho^c_{n+1}
\end{array} \right).$$

Before closing this section, we just note that instead of precisely calculated values of scaling factors ρ several linear programming codes usually use the nearest powers of two as a 'binary approximation' of these values. The reason is that for computers based on the binary system, it may dramatically improve performance of scaling since in this case the relatively expensive operation of multiplication may be implemented as very fast shifting of data to the left or right, depending on the power of 2 used for such 'approximation'.

4. Numeric example

In the previous section we considered three rules for calculating scaling factors[2]. All three rules are suitable to be used for automatic scaling in programming packages and allow relatively easy achievement of a well-scaled problem.

To illustrate how these scaling factors work, we consider the following rectangular matrix of order 7×5

$$A = \begin{pmatrix} 0.0005 & 3.000 & 0.340 & 234.000 & 34.000 \\ 2.0000 & 4.000 & 345.000 & 1234.000 & 234.000 \\ 30000.0000 & 5.000 & 4565643.000 & 34.000 & 234.000 \\ 9.0000 & 6.000 & 0.001 & 567.000 & 4.000 \\ 567.0000 & 7.000 & 234.000 & 24.000 & 234.000 \\ 56.0000 & 8.000 & 345.000 & 0.001 & 3.000 \\ 45000.0000 & 9.000 & 4.000 & 3.000 & 123.000 \end{pmatrix}. \tag{20}$$

This matrix may be said to be **badly scaled** since

$$\begin{aligned} \max_{i,j \in J_+}(|a_{ij}|) = a_{33} &= 4565643.000 = 4.565643E+06, \\ \min_{i,j \in J_+}(|a_{ij}|) = a_{11} &= 0.0005 = 5.000000E-04; \end{aligned}$$

and

$$\sigma(A) = \frac{\max\limits_{i,j \in J_+}(|a_{ij}|)}{\min\limits_{i,j \in J_+}(|a_{ij}|)} = \frac{4.565643E+06}{5.00E-04} = 9.13E+09,$$

i.e. the magnitude between the largest and the smallest absolute values of non-zero entries a_{ij} is of order 10 ($\sigma(A) = 9.13E+09 \approx 1.0E+10$).

[2]The first two rules have been implemented in the linear programming codes developed at Edinburgh University, Department of Mathematics and Statistics, Scotland. Mean-rule is used in the package developed by J.Gondzio for sparse and dense large-scale linear programming; the package implements some special algorithm of the method of interior point. Geometric-rule is implemented in the package of J.A.J.Hall for very sparse large-scale linear programming problems; the package is based on the revised simplex method. Max/Min-rule has been tested in the linear and linear-fractional parallel package PGULF developed by E.Bajalinov during his research visit at Edinburgh Parallel Computing Centre, Edinburgh University, Scotland.

4.1. **Mean rule.** First, we apply successively Mean rule factors for rows and columns to scale matrix A. The results of scaling are as follows.

Original matrix: In accordance with rule (18), vector ρ^r of row scaling factors for original matrix A is

$$\rho^r = (0.3421, 49.6790, 4777.8881, 0.7530, 63.0000, 0.5874, 367.4235)^T.$$

Perform row scaling.

After 1st scaling of rows: For modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j\in J_+}(|a_{ij}|) \;=\; 9.56E+02; \quad \min_{i,j\in J_+}(|a_{ij}|) \;=\; 1.05E-03;$$

$$\sigma(A) \;=\; \frac{9.56E+02}{1.05E-03} \;=\; 9.13E+05.$$

We use rule (19) to calculate vector ρ^c of column scaling factors:

$$\rho^c = (0.4231, 0.1194, 1.1265, 1.1322, 2.2064).$$

Perform column scaling.

After 1st scaling of columns: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j\in J_+}(|a_{ij}|) \;=\; 8.48E+02; \quad \min_{i,j\in J_+}(|a_{ij}|) \;=\; 1.18E-03;$$

$$\sigma(A) \;=\; \frac{8.48E+02}{1.18E-03} \;=\; 7.20E+05.$$

Vector ρ^r of row scaling factors:

$$\rho^r = (1.4448, 1.4448, 2.3090, 0.8854, 2.6752, 0.8854, 1.4448)^T.$$

Perform row scaling.

After 2nd scaling of rows: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j\in J_+}(|a_{ij}|) \;=\; 7.51E+02; \quad \min_{i,j\in J_+}(|a_{ij}|) \;=\; 1.33E-03;$$

$$\sigma(A) \;=\; \frac{7.51E+02}{1.33E-03} \;=\; 5.64E+05.$$

Vector ρ^c of column scaling factors:

$$\rho^c = (0.7801, 0.6994, 0.8854, 1.1294, 0.5475).$$

Perform column scaling.

After 2nd scaling of columns: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j\in J_+}(|a_{ij}|) \;=\; 6.65E+02; \quad \min_{i,j\in J_+}(|a_{ij}|) \;=\; 1.50E-03;$$

$$\sigma(A) \;=\; \frac{6.65E+02}{1.50E-03} \;=\; 4.42E+05.$$

Vector ρ^r of row scaling factors:

$$\rho^r = (1.0654, 1.0654, 1.0000, 1.0000, 1.0654, 1.0000, 1.0654)^T.$$

Perform row scaling.

After 3rd scaling of rows: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j \in J_+}(|a_{ij}|) \;=\; 6.65E+02; \quad \min_{i,j \in J_+}(|a_{ij}|) \;=\; 1.50E-03;$$

$$\sigma(A) \;=\; \frac{6.65E+02}{1.50E-03} \;=\; 4.42E+05.$$

Vector ρ^c of column scaling factors:

$$\rho^c = (0.9688, 1.0000, 1.0000, 1.0000, 0.9688).$$

After performing multiple successive scaling operations for rows and columns, we obtain scaling factors both for rows and columns with values close to 1. Hence, there is no reason to continue this process, since the further improvement of ill-scaling measure $\sigma(A)$ for matrix A becomes more and more expensive.

So, starting from the original matrix A with $\sigma(A) = 9.13E+09$ we obtained its scaled modification with $\sigma(A) = 4.42E+05$. As we can see, the improvement of magnitude achieved is of order $5 = 10 - 5$.

4.2. **Geometric rule.** Now, let us apply Geometric rule factors to scale the same matrix A given in (20). We have the following results.

Original matrix:: In accordance with rule (16) we calculate vector ρ^r of row scaling factors

$$\rho^r = (1.3233, 60.2959, 1403.6460, 2.6158, 87.7940, 3.4138, 56.9257)^T.$$

Perform row scaling.

After 1st scaling of rows: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j \in J_+}(|a_{ij}|) \;=\; 3.25E+03; \quad \min_{i,j \in J_+}(|a_{ij}|) \;=\; 2.93E-04;$$

$$\sigma(A) \;=\; \frac{3.25E+03}{2.93E-04} \;=\; 1.11E+07.$$

Row-vector ρ^c of column scaling factors calculated in accordance with rule (17)

$$\rho^c = (1.8606, 0.2321, 1.6591, 0.6973, 2.0014).$$

Perform column scaling.

After 1st scaling of columns: For the modified matrix we calculate measure $\sigma(A)$ of ill-scaling:

$$\max_{i,j\in J_+}(|a_{ij}|) \;=\; 1.96E+03; \quad \min_{i,j\in J_+}(|a_{ij}|) \;=\; 2.03E-04;$$

$$\sigma(A) \;=\; \frac{1.96E+03}{2.03E-04} \;=\; 9.65E+06.$$

Column-vector ρ^r of row scaling factors will be

$$\rho^r = (1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000)^T.$$

Moreover, rule (17) used to calculate new row-vector ρ^c of column scaling factors gives

$$\rho^c = (1.0000, 1.0000, 1.0000, 1.0000, 1.0000).$$

After performing two successive scaling operations for rows and columns, we obtain scaling factors both for rows and columns with values exactly equal to 1. Hence, there is no reason to continue this process, since the further improvement of ill-scaling measure $\sigma(A)$ for matrix A using this rule is impossible.

So, starting from the original matrix A with $\sigma(A) = 9.13E+09$ we obtained its scaled modification with $\sigma(A) = 9.65E+06$. As we can see, the improvement of magnitude achieved is of order $4 = 10-6$.

4.3. **Max/Min-factors.** Finally, let us apply Max/min-factors to scale the same matrix A given in (20).

First, we define the smallest entry $a_{11} = 0.0005$ and the largest entry $a_{33} = 4565643.000$, then in the row 1 (that contains the smallest entry) we find the largest entry $a_{14} = 234.00$ and calculate factor ρ'_r for row 1 as follows

$$\rho'_r \;=\; \frac{\max\limits_j |a_{1j}| + a_{33}}{2\times \max\limits_j |a_{1j}|} \;=\; \frac{234.00 + 4565643.000}{2\times 234.00} = 9756.1475.$$

Using the factor obtained we scale row 1 and obtain the following

Row 1 = 4.8781 29268.4414 3317.0901 2282938.5 331709.000

Now, the smallest non-zero entry of the matrix is $a_{43} = 0.001$, so

$$\sigma(A) \;=\; \frac{\max\limits_{i,j\in J_+}(|a_{ij}|)}{\min\limits_{i,j\in J_+}(|a_{ij}|)} \;=\; \frac{4.565643E+06}{1.00E-03} \;=\; 4.565643E+09,$$

Note, that in the scaled matrix the smallest entry $a_{43} = 0.001$ is in the same column 3 as the largest entry $a_{33} = 4565643.00$, so there is no reason for scaling this column since in this case the value for scaling factors ρ'_c and ρ''_c will be exactly 1. Indeed,

$$\rho'_c = \frac{\max\limits_i |a_{i3}| + a_{33}}{2 \times \max\limits_i |a_{i3}|} = \frac{a_{33} + a_{33}}{2 \times a_{33}} = 1,$$

and

$$\rho''_c = \frac{\min\limits_{i:\, a_{i3} \neq 0} |a_{i3}| + a_{43}}{2 \times \min\limits_{i:\, a_{i3} \neq 0} |a_{i3}|} = \frac{a_{43} + a_{43}}{2 \times a_{43}} = 1.$$

This is why in the next scaling operation we have to scale a row. Arbitrarily, we choose the row of the largest entry and calculate the following factor

$$\rho''_r = \frac{\min\limits_{j:\, a_{3j} \neq 0} |a_{3j}| + a_{43}}{2 \times \min\limits_{j:\, a_{3j} \neq 0} |a_{3j}|} = \frac{a_{32} + a_{43}}{2 \times a_{32}} = \frac{5.00 + 0.001}{2 \times 5.00} = 0.5001$$

Performing the scaling we obtain

$$\text{Row } 3 = 15003.0000\ 2.5005\ 2283278.0643\ 17.0034\ 117.0234$$

and new value for the measure of ill-scaling

$$\sigma(A) = \frac{\max\limits_{i,j \in J_+} (|a_{ij}|)}{\min\limits_{i,j \in J_+} (|a_{ij}|)} = \frac{2283278.0643}{0.0010} = 2.28328E + 09,$$

so the resulting matrix is as follows

$$A = \begin{pmatrix} 4.8781 & 29268.4414 & 3317.0901 & 2282938.5 & 331709.000 \\ 2.0000 & 4.000 & 345.000 & 1234.000 & 234.000 \\ 15003.0000 & 2.5005 & 2283278.0643 & 17.0034 & 117.0234 \\ 9.0000 & 6.000 & 0.001 & 567.000 & 4.000 \\ 567.0000 & 7.000 & 234.000 & 24.000 & 234.000 \\ 56.0000 & 8.000 & 345.000 & 0.001 & 3.000 \\ 45000.0000 & 9.000 & 4.000 & 3.000 & 123.000 \end{pmatrix}.$$

Note, that the largest and the smallest entries of the matrix are in the same column, so in the next scaling operation we have to scale a row. After performing 9 such scaling operations we obtain resulting matrix with $\sigma(A) = 4.92E + 05$.

Observe, that when using this scaling rule in each step we scale only one row or column, so the rule is not so expensive as the previous two scaling rules.

5. Testing and comparison of the scaling rules

In this section we briefly describe the main results obtained during testing the scaling rules mentioned above. A special testing program was developed in C++. The most important properties we focused on are as follows:

- How long time (or iterations) does the rule take to produce a predefined improvement.
- How long time (or iterations) does the rule take to produce resulting matrix with a predefined $\sigma(A)$;
- How great an improvement can be achieved in one scaling iteration;

We tested these rules using 1100 randomly generated matrices with a size from 10×10 to 130×130 with different densities. For each size there were generated 9 matrices with different density and performed all three scaling rules.

The main results obtained may be summarized as follows:

(1) All three rules can produce resulting matrix with a better $\sigma(A)$.
(2) When applying Geometric-rule or Mean-rule after several (depending on the size and the density of matrix) iterations scaling factors for rows and columns tend to 1.0, so no further improvement can be achieved. Moreover, in each scaling iteration we have to scale successively all rows and all columns. The best improvement was achieved for matrices with relatively small size and the density between 20% and 40%.
(3) Max/Min-rule in each iteration requires scaling only one row or column and usually produces a better matrix. At the same time, the more iterations, the slower improvement.

Summarizing our test results we present the following two tableaus. The first of them shows the (average) improvement achieved by different methods for the problems of different density. The second one presents the same results grouped by the size of the test problems.

Geometric rule		Mean rule		Min/Max rule	
Sparsity	Improvement	Sparsity	Improvement	Sparsity	Improvement
0%	7,93E-01	0%	4,24E+01	0%	4,42E+01
10%	4,58E-01	10%	3,34E+01	10%	2,33E+01
20%	8,61E-01	20%	6,76E+01	20%	2,65E+01
30%	3,86E-01	30%	6,62E+01	30%	5,82E+01
40%	6,62E-01	40%	4,79E+01	40%	6,56E+01
50%	5,14E+00	50%	3,11E+01	50%	2,67E+01
60%	5,04E-01	60%	5,15E+01	60%	2,76E+01
70%	4,37E-01	70%	5,32E+01	70%	2,86E+01
80%	4,46E-01	80%	4,15E+01	80%	2,55E+01

Geometric rule		Mean rule		Min/Max rule	
Probl. Size nxn	Improvement	Probl. Size nxn	Improvement	Probl. Size nxn	Improvement
100	5,31E+00	100	1,65E+02	100	4,03E+02
256	1,25E+00	256	3,26E+01	256	2,68E+02
484	4,18E-01	484	5,37E+01	484	7,67E+01
784	5,27E-01	784	2,74E+01	784	6,71E+01
1156	6,91E-01	1156	1,02E+02	1156	2,16E+01
1600	3,62E-01	1600	5,94E+01	1600	2,21E+01
2116	5,11E-01	2116	2,70E+01	2116	1,99E+01
2704	2,91E-01	2704	1,07E+02	2704	2,04E+01
3364	2,74E-01	3364	2,67E+01	3364	3,12E+01
4096	3,62E-01	4096	2,86E+01	4096	3,95E+01
4900	2,86E-01	4900	5,01E+01	4900	1,75E+01
5776	3,54E-01	5776	3,36E+01	5776	2,15E+01
6724	3,52E-01	6724	3,15E+01	6724	2,08E+01
7744	2,61E-01	7744	9,86E+01	7744	1,32E+01
8836	5,14E-01	8836	2,03E+01	8836	3,35E+01
10000	4,21E-01	10000	5,36E+01	10000	2,67E+01
11664	4,52E-01	11664	4,14E+01	11664	1,20E+01
13456	3,01E-01	13456	2,16E+01	13456	1,58E+01
15376	3,49E-01	15376	6,88E+01	15376	1,45E+01
16900	3,93E-01	16900	3,49E+01	16900	2,05E+01

All tests were performed in the following environment:

- Operating system: Microsoft Windows XP Home Edition +SP2
- CPU: Mobile AMD Turion 64 MT-32, 1.8GHz
- RAM: 512Mb

References

[1] E. B. Bajalinov. *Linear-fractional programming. Theory, methods, applications and software*. Kluwer Academic Publishers, 2003.

[2] A. Curtis and J. Reid. On the automatic scaling of matrices for Gaussian elimination. *J. Inst. Math. Appl.*, 10:118–124, 1972.

[3] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct methods for sparse matrices*. Monographs on Numerical Analysis. The Clarendon Press Oxford University Press, New York, 1986. Oxford Science Publications.

[4] S. Pissanetzky. *Sparse matrix technology*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, 1984.

[5] R. D. Skeel. Scaling for numerical stability in Gaussian elimination. *J. Assoc. Comput. Mach.*, 26:494–526, 1979.

Received on May 12, 2009, accepted on June 6, 2009

Faculty of Informatics,
Debrecen University,
4010 Debrecen, Egyetem tér 1, P.O.B. 12, Hungary
E-mail address: bajalinov.erik@inf.unideb.hu
E-mail address: racz.anett@inf.unideb.hu