Columbia University
Columbia University Biostatistics Technical Report Series

Year 1985 *Paper* 10

On Calculating Maximum Rank One Underapproximations for Positive Arrays

Bruce Levin*

*Columbia University, Bruce.Levin@Columbia.edu

On Calculating Maximum Rank One Underapproximations for Positive Arrays

Bruce Levin

Abstract

Given P, a rectangular array with positive elements, a rank one underapproximation for P is given by two positive vectors, say r and s, such that each component of rs' is no greater than the corresponding component of P, whence P can then be written as P=(pi)rs'+(1-pi)D for some constant pi, where the residual matrix D is non-negative. A maximal rank one underapproximation for P is such that pi is maximized over all possible rank one underapproximations for P.

We provide an algorithm for calculating the maximum rank one underapproximation and corresponding pi. We present an explicit expression in the special case of 2 x c tables, and show that the algorithm yields the correct solution in this case.

Note: this report was originally entitled "Technical Report No. B-48, January 1985" in the Columbia Biostatistics Tech Report Series. Due to early font styles, strict and non-strict inequalities may be difficult to distinguish in the scanned version. Magnifying the typeface will help.

TECHNICAL REPORT NO. B-48
JANUARY 1985

BRUCE LEVIN

On calculating maximum rank one underapproximations for positive arrays.

Given: A positive matrix $\{P_{ij}\}$ of dimensions $I \times J$ with $p_{ij} > 0$ and $\sum_{ij} p_{ij} = 1$.

Problem: Find vectors $r = (r_1,\ldots,r_I)'$ and $s = (s_1,\ldots,s_J)'$ with $0 < r_i, s_j < 1$ and $\sum r_i = \sum s_j = 1$ such that $P = \pi rs' + (1-\pi) D$ for some $0 < \pi < 1$ and $D = \{d_{ij}\}$ $(d_{ij} \geq 0)$, with maximum possible π .

Equivalent problem: Find vectors $R = (R_1,\ldots,R_I)'$ and $S = (S_1,\ldots,S_J)'$ with $\sum R_i = 1$, $R_i > 0$, $S_j > 0$ such that $\sum S_j$ is maximized subject to $p_{ij} \geq R_i S_j$ for all (i,j). [For the original problem take $r = R$, $s = S / \sum S_j$, $\pi = \sum S_j$.]

Formulation as a convex-programming problem: Let $x_i = \log R_i$ $(i = 2,\ldots,I)$ and $y_j = \log S_j$ $(j = 1,\ldots,J)$, $x = (x_2,\ldots,x_I)$, $y = (y_1,\ldots,y_J)$.

$$\text{Maximize} \quad f(x,y) = \sum_1^J \exp(y_j)$$

subject to

$$g_0(x,y) = 1 - \sum_2^I \exp(x_i) \geq 0$$

and for $j = 1,\ldots,J$,

$$g_{1j}(x,y) = \log p_{1j} - \log \left(1 - \sum_2^I \exp(x_i)\right) - y_j \geq 0$$

$$g_{ij}(x,y) = \log p_{ij} - x_i - y_j \geq 0 \quad (i > 1).$$

There are $IJ + 1$ constraints; g_0 is concave, g_{1j} are convex, g_{ij} are linear for $i > 1$. The Kuhn-Tucker necessary conditions for optimality of a solution (x,y) are that

(i) $g_0(x,y) \geq 0$, $g_{ij} \geq 0$ all (i,j); [(x,y) feasible]

(ii) There exist $\lambda_0, \lambda_{ij} \geq 0$ such that $\lambda_0 g_0(x,y) = 0$, $\lambda_{ij} g_{ij}(x,y) = 0$;
[inactive constraints require zero multipliers]

(iii) $\nabla f(x,y) + \lambda_0 \nabla g_0(x,y) + \sum_{ij} \lambda_{ij} \nabla g_{ij}(x,y) = 0$.
[negative gradient of f lies within cone formed by active constraint gradients]

It is clear that any optimal solution will not have g_0 as an active constraint, i.e. $R_1 > 0$, for given any feasible (x,y) with corresponding $R_1 = 0$, we can decrease $R_2,\ldots,R_I$ slightly and increase R_1 up to $\min_j (p_{1j}/S_j)$, preserving the constraints $p_{ij} \geq R_i S_j$, and leaving $\sum S_j$ unchanged. Thus $\lambda_0 = 0$. In fact the Kuhn-Tucker necessary conditions reduce to the following: if (R,S) is an optimal solution, then for each cell (i,j) with an active constraint $p_{ij} = R_i S_j$ there is a $\lambda_{ij} \geq 0$ such that, putting $\lambda_{ij} = 0$ for any cell with $p_{ij} > R_i S_j$,

$$\lambda_{+j} = \sum_{i=1}^{I} \lambda_{ij} = S_j \qquad (j = 1,\dots,J)$$

and

$$\frac{\lambda_{1+}}{R_1} = \dots = \frac{\lambda_{I+}}{R_I} = \text{constant.}$$

(The constant equals $\sum S_j$.) There are $I + J - 1$ equations so that if the linear system in non-singular, there must be at least $I + J - 1$ active constraints.

Unfortunately the Kuhn-Tucker conditions are not generally sufficient conditions useful for finding the optimal solution. Below we give a simple computing algorithm for calculating an optimal solution that does not require explicit solutions of the constraint equations or Lagrange multipliers. I conjecture, but have not proved, that the algorithm always terminates in finite time with a globally maximum rank one underapproximation to P. In the special case of 2 x J tables, a closed form solution is available, and this result is presented below. Our algorithm produces the optimal solution in this case.

Algorithm. For efficiency, arrange the table so that $I > J$.

[0] Initialization: An ordered list of the $K = 2^J - 1$ non-empty subsets $B_k \subseteq \{1,\dots,J\}$ is produced; for large K an alternative is to replace the list with an algorithm for generating $B_{1+k \bmod K}$ from B_k for $k = 1,\dots,K$.

For n=1,...,J, set: $R_i^{(0,n)} = p_{in}/p_{+n}$ $(i = 1,\ldots,I)$

$S_j^{(0,n)} = p_{+n} \cdot \min_{1\le i\le I} p_{ij}/p_{in}$ $(j = 1,\ldots,J)$

and

$\pi^{(0,n)} = \Sigma_j\ S_j^{(0,n)}.$

These quantities are used as J different sets of initial values as n=1,...,J, and steps [1]-[5] below are carried out for each set. For any choice of n, put $k \leftarrow \nu \leftarrow c \leftarrow 0$.

[1] Set $k \leftarrow 1 + k \bmod K$

$B \leftarrow B_k$.

[2] Define $A = A_k^{(\nu)} = \{i: p_{ij} = R_i^{(\nu)} S_j^{(\nu)}$ for some $j \in B_k\}$.

[nb: $A_k^{(\nu)} \ne \phi$ all ν,k .]

If $A = \{1,\ldots,I\}$ then go to [3] else define:

$$m = m_k^{(\nu)} = \min_{i\epsilon A^c,\ j\epsilon B} \{p_{ij} / R_i^{(\nu)} S_j^{(\nu)}\}$$

$$\rho = \rho_k^{(\nu)} = \Big(\sum_{i\epsilon A} R_i^{(\nu)}\Big) / \Big(\sum_{i\epsilon A^c} R_i^{(\nu)}\Big)$$

$$\sigma = \sigma_k^{(\nu)} = \Big(\sum_{j\epsilon B} S_j^{(\nu)}\Big) / \Big(\sum_{j\epsilon B^c} S_j^{(\nu)}\Big) .$$

If $m\sigma > \rho$ then go to [4] else go to [3].

[nb: The condition may be rewritten to avoid the value ∞ for σ when

$B = \{1,\ldots,J\}$ as follows:

$$m \sum_{i \varepsilon A^c} R_i^{(\nu)} \cdot \sum_{j \varepsilon B} S_j^{(\nu)} > \sum_{i \varepsilon A} R_i^{(\nu)} \cdot \sum_{j \varepsilon B^c} S_j^{(\nu)} .]$$

[3] Set $c \leftarrow c + 1$. If $c = K$ then <u>stop</u> else go to [1].

[4] Set $c \leftarrow 0$. Define $\tau = \tau_k^{(\nu)} = \frac{m-1}{m+\rho}$ so that $m = \frac{1+\tau\rho}{1-\tau}$.

$$\text{Set } R_i^{(\nu+1)} \leftarrow \begin{cases} R_i^{(\nu)} \cdot (1-\tau) & (i \;\varepsilon\; A) \\ R_i^{(\nu)} \cdot (1+\tau\rho) & (i \;\varepsilon\; A^c) \end{cases} \qquad (i = 1,\ldots,I)$$

$$S_j^{(\nu+1)} \leftarrow \begin{cases} S_j^{(\nu)} / (1-\tau) & (j \;\varepsilon\; B) \\ S_j^{(\nu)} / (1+\tau\rho) & (j \;\varepsilon\; B^c) \end{cases} \qquad (j = 1,\ldots,J) .$$

[5] For $j = 1,\ldots,J$ set

$$S_j^{(\nu+1)} \leftarrow S_j^{(\nu+1)} \cdot \min_{1 \leq i \leq I} \{p_{ij} / (R_i^{(\nu+1)} S_j^{(\nu+1)})\} = \min_{1 \leq i \leq I} \{p_{ij} / R_i^{(\nu+1)}\} .$$

[nb: This only affects j not in B.]

Set $\pi^{(\nu+1)} \leftarrow \sum_{j=1}^{J} S_j^{(\nu+1)}$,

$\nu \leftarrow \nu + 1$, and

go to [2].

Principle of algorithm:

–	–	–	=		–	–	–	
...	–	...	=	=	...	–	...	$\cdot\ (1-\tau) \quad (i \in A)$
–	–	–		=	–	–	–	
			+	+				$\cdot\ (1+\tau\rho) \quad (i \in A^C)$
			+	+				
$\div(1+\tau\rho)$ $(j \in B^C)$			$\div(1-\tau)$ $(j \in B)$		$\div(1+\tau\rho)$ $(j \in B^C)$			

(a) Cells marked $=$ are such that

$$P_{ij} = R_i^{(\nu)} S_j^{(\nu)} = R_i^{(\nu+1)} S_j^{(\nu+1)} \quad \text{for} \quad j \in B \ ;$$

cells marked $+$ increase as a result of [4],

cells marked $-$ decrease as a result of [4], and

unmarked cells are unchanged. Step [5] ensures that at least one constraint is active in each column.

(b) The definition of ρ ensures that $\sum_i R_i^{(\nu+1)} = \sum_i R_i^{(\nu)} = 1$.

(c) $\pi^{(\nu+1)} = \sum_j S_j^{(\nu+1)} > \sum_j S_j^{(\nu)} = \pi^{(\nu)}$.

Proof: As a result of [4],

$$\sum_j S_j^{(\nu+1)} = \sum_{j \in B^C} S_j^{(\nu)} / (1 + \tau\rho) + \sum_{j \in B} S_j^{(\nu)} / (1 - \tau)$$

$$> \sum_{j \in B^C} S_j^{(\nu)} + \sum_{j \in B} S_j^{(\nu)} = \sum_j S_j^{(\nu)}$$

if and only if

$$\sigma\left(\frac{1}{1-\tau}-1\right) > \left(1-\frac{1}{1+\tau\rho}\right) \quad \text{if and only if} \quad \sigma(1+\tau\rho) > \rho(1-\tau)$$

which is $m\sigma > \rho$.

Step [5] can only further increase $\sum_j S_j^{(\nu+1)}$.

(d) Examples show that the terminal value of π depends on the initial values of $R_i{}^{(0)}$. Hence what needs to be proven is that for at least one choice of $R^{(0,n)}$ the set of directions defined by the algorithm for movement of R in the I-simplex is sufficiently rich so that no further improvement is possible at any stage ($m\sigma \le \rho$ for all choices B_k, $k = 1,\ldots,K$) only when $\pi^{(\nu+1)}$ has reached the global maximum.

Explicit solution for 2 x J tables.

Arrange the columns of P such that $(p_{11} / p_{21}) > \ldots > (p_{1J} / p_{2J})$.

Theorem The maximum achievable π is $\pi^* = \max(\pi_1,\ldots,\pi_J)$ where

$$\pi_j = 1 - \sum_{k<j} p_{1k}(1-\omega_{kj}^{-1}) - \sum_{k>j} p_{2k}(1-\omega_{jk}^{-1})$$

and

$\omega_{jk} = \dfrac{p_{1j} / p_{2j}}{p_{1k} / p_{2k}}$ is the odds ratio from columns (j,k); $\omega_{jk} > 1$ for $j < k$.

The solution $r_1 = R_1 = 1 - R_2 = 1 - r_2 = p_{1g} / (p_{1g} + p_{2g})$ where g is such that

$$\pi_g = \max\{\pi_1,\ldots,\pi_J\}$$

and

$$s_j = S_j / \pi^* = \begin{cases} \left(1 + \sum_{k<g} (p_{2k} / p_{2g}) + \sum_{k>g} (p_{1k} / p_{1g})\right)^{-1} & (j = g) \\ s_g \, (p_{1j} / p_{1g}) & (j > g) \\ s_g \, (p_{2j} / p_{2g}) & (j < g) . \end{cases}$$

The constraints are active $(p_{ij} = R_i S_j)$ for $i = 1, j > g$ and $i = 2, j < g$.

Proof. Clearly, $\pi < \frac{p_{ij}}{r_i s_j}$ for all i,j so that $\pi < \min_{i,j} \{\frac{p_{ij}}{r_i s_j}\}$ and it suffices to evaluate

$$\pi^* = \max_{r,s} \min \left\{ \frac{1}{r_1} \min \left(\frac{p_{11}}{s_1}, \ldots, \frac{p_{1J}}{s_J} \right), \frac{1}{r_2} \min \left(\frac{p_{21}}{s_1}, \ldots, \frac{p_{2J}}{s_J} \right) \right\} .$$

For $i = 1,2$ define the function $m_i(s) = \min \{ \frac{p_{i1}}{s_1}, \ldots, \frac{p_{iJ}}{s_J} \}$ for s in the J-simplex. For each i, the simplex is partitioned into J regions U_{ij} $(j = 1,\ldots,J)$ in which $m_i(s) = p_{ij} / s_j$. The regions U_{ij} are convex, and are bounded by the simplex boundaries and hyperplanes of the form $\{ \frac{p_{ij}}{s_j} = \frac{p_{ij'}}{s_{j'}} \}$. Consider also the $J(J + 1) / 2$ intersections $V_{jk} = U_{1j} \& U_{2k}$ for $j > k$; for $j < k$ the interior of V_{jk} is empty by the standardization $(p_{11} / p_{21}) > \ldots > (p_{1J} / p_{2J})$, and so may be ignored.

Lemma 1. The maximum of $m_i(s)$ on V_{jj} occurs at a point $s^{(j)}$ satisfying the equations

$$\{ \frac{p_{1j}}{s_j} = \ldots = \frac{p_{1J}}{s_J} \} \text{ and } \{ \frac{p_{21}}{s_1} = \ldots = \frac{p_{2j}}{s_j} \} .$$

$\underline{\text{Proof}}$. Let $k < j$. Then

$$\frac{s_k^{(j)}}{s_j^{(j)}} = \frac{p_{2k}}{p_{2j}} < \frac{p_{1k}}{p_{1j}} \text{ implies } s^{(j)} \in U_{1j} \text{ ; similarly } s^{(j)} \in U_{2j} .$$

Hence $s^{(j)} \in V_{jj}$. If $t \in V_{jj}$ were such that $\frac{p_{1j}}{t_j} > \frac{p_{1j}}{s_j^{(j)}}$, then for some component $k \neq j$, $t_k > s_k^{(j)}$. But k cannot equal any $j, j+1, \ldots, J$ for otherwise

$$\frac{p_{1j}}{t_j} > \frac{p_{1j}}{s_j^{(j)}} = \frac{p_{1k}}{s_k^{(j)}} > \frac{p_{1k}}{t_k}$$

contradicting $m_1(t) = p_{1j} / s_j$. Nor can $k = 1, \ldots, j$ for then

$$\frac{p_{2k}}{t_k} < \frac{p_{2k}}{s_k^{(j)}} = \frac{p_{2j}}{s_j^{(j)}} < \frac{p_{2j}}{t_j}$$

contradicting $m_2(t) = p_{2j} / t_j$. Thus for any $t \in V_{jj}$,

$$m_1(t) = \frac{p_{1j}}{t_j} < \frac{p_{1j}}{s_j^{(j)}} = m_1(s^{(j)}) ,$$

proving the lemma. [nb: $s^{(j)}$ is given explicitly by

$$s_k^{(j)} = \begin{cases} s_j^{(j)} \dfrac{p_{1k}}{p_{1j}} & (k > j) \\[2ex] s_j^{(j)} \dfrac{p_{2k}}{p_{2j}} & (k < j) . \end{cases}$$

Thus $s_j^{(j)} = \left(1 + \sum_{k<j} \frac{p_{2k}}{p_{2j}} + \sum_{k>j} \frac{p_{1k}}{p_{1j}} \right)^{-1}$.]

Lemma 2. All extreme points of V_{jk} $(j > k)$ not on the simplex boundary are contained among the $s^{(j)}$ $(j = 1,\ldots,J)$.

Proof. Any extreme point s of V_{jk} not on the simplex boundary is given by a set of $J - 1$ equalities of the form

$$\frac{p_{1j_0}}{s_{j_0}} = \frac{p_{1j_1}}{s_{j_1}} = \cdots = \frac{p_{1j_m}}{s_{j_m}} \text{ and } \frac{p_{2k_0}}{s_{k_0}} = \frac{p_{2k_1}}{s_{k_1}} = \cdots = \frac{p_{2k_n}}{s_{k_n}}$$

for some $m,n \geq 0$ with $m + n = J - 1$ and where $j_0 = j$, $k_0 = k$. If $m = 0$ (resp. $n = 0$) then $s = s^{(J)}$ (resp. $s = s^{(1)}$). Suppose $m,n > 0$. Since there are $(m + 1) + (n + 1) = J + 1$ variables among the $J - 1$ equalities, we must have $j_\alpha = k_\beta$ for some α,β . But there can be no other matching subscripts $j_{\alpha'} = k_{\beta'}$ for otherwise the equations are inconsistent, if $(p_{1j_\alpha} / p_{2k_\beta}) \neq (p_{1j_{\alpha'}} / p_{2k_{\beta'}})$, or else redundant (if equality occurs). In either case an extreme point is not obtained. We proceed to show that s must equal $s^{(j)}$ for some j . Call the common value $j_\alpha = k_\beta = \ell$. Suppose $j_i < \ell$. Then

$$\frac{p_{1j_i}}{s_{j_i}} = \frac{p_{1\ell}}{s_\ell} \text{ implies } \frac{s_{j_i}}{s_\ell} = \frac{p_{1j_i}}{p_{1\ell}} \geq \frac{p_{2j_i}}{p_{2\ell}}$$

$$\text{implies } \frac{p_{2j_i}}{s_{j_i}} \leq \frac{p_{2\ell}}{s_\ell} = \frac{p_{2k_0}}{s_{k_0}} = m_2(s) \text{ implies } \frac{p_{2j_i}}{s_{j_i}} = \frac{p_{2\ell}}{s_\ell} .$$

Similarly $\frac{p_{1k_i}}{s_{k_i}} = \frac{p_{1\ell}}{s_\ell}$ for $k_i > \ell$. Thus

$$\frac{p_{1\ell}}{s_\ell} = \cdots = \frac{p_{1J}}{s_J} \text{ and } \frac{p_{21}}{s_1} = \cdots = \frac{p_{2\ell}}{s_\ell} \text{ which means } s = s^{(\ell)} ,$$

proving the lemma.

Returning to the proof of the theorem, to maximize the convex functions $m_1(s)$, $m_2(s)$, or $m_1(s) + m_2(s)$ over the convex regions V_{jk}, from the lemmas it suffices to maximize over the points $s^{(j)}$ $(j = 1,\ldots,J)$. Now

$$\min \left\{ \frac{1}{r_1} m_1(s) , \frac{1}{1-r_1} m_2(s) \right\} = \begin{cases} m_1(s) / r_1 & \text{if } r_1 > \dfrac{m_1(s)}{m_1(s) + m_2(s)} \\ m_2(s) / r_2 & \text{if } r_1 < \dfrac{m_1(s)}{m_1(s) + m_2(s)} \end{cases}$$

for fixed s, so that $\max\limits_{r} \min \left\{ \frac{1}{r_1} m_1(s) , \frac{1}{r_2} m_2(s) \right\} = m_1(s) + m_2(s)$ and is achieved for $r_1 = m_1(s) / [m_1(s) + m_2(s)]$. It remains only to maximize over s as indicated above:

$$\pi^* = \max_s \left[m_1(s) + m_2(s) \right] = \max_j \left[m_1(s^{(j)}) + m_2(s^{(j)}) \right] = \max_j \left[\frac{p_{1j} + p_{2j}}{s_j^{(j)}} \right]$$

since $s^{(j)} \varepsilon V_{jj}$

$$= \max_j \left[(p_{1j} + p_{2j}) \left(1 + \sum_{k<j} \frac{p_{2k}}{p_{2j}} + \sum_{k>j} \frac{p_{1k}}{p_{1j}}\right) \right] = \max_j \pi_j$$

where $\pi_j = 1 - \sum\limits_{k<j} p_{1k}(1 - \omega_{kj}^{-1}) - \sum\limits_{k>j} p_{2k}(1 - \omega_{jk}^{-1})$.

If $\pi_g = \max\limits_j \pi_j$, then $s = s^{(g)}$, and $r_1 = m_1(s^{(g)}) / [m_1(s^{(g)}) + m_2(s^{(g)})]$ $= p_{1g} / (p_{1g} + p_{2g})$. Finally, if $j > g$ then

$$p_{1j} - \pi^* r_1 s_j = p_{1j} - \left(\frac{p_{1g} + p_{2g}}{s_g^{(g)}}\right) \frac{p_{1g}}{(p_{1g} + p_{2g})} \frac{p_{1j}}{p_{1g}} s_g^{(g)} = 0 ,$$

and similarly for $p_{2j} - \pi^* r_2 s_j$ $(j < g)$. Q.E.D.

Remark It is clear that when our numerical algorithm is applied in the 2 x J case (with transposing in step [0]), the optimal values will be found, since the list of initial values $R_1^{(0,n)}$ for n = 1,...,J contains $p_{1g} / (p_{1g} + p_{2g})$. When applied to the transposed J x 2 table, the algorithm will either start at or move to one of the J points $s^{(j)}$ in the J-simplex. In either case the algorithm finds π^*.